


Article

# A Novel Cooperative Path Planning for Multi-robot Persistent Coverage with Obstacles and Coverage Period Constraints

Guibin Sun <sup>1,\*</sup> , Rui Zhou <sup>1</sup>, Bin Di <sup>2</sup>, Zhuoning Dong <sup>1</sup> and Yingxun Wang <sup>1</sup>

<sup>1</sup> School of Automation Science and Electrical Engineering, Beihang University, Beijing 100083, China; zhr@buaa.edu.cn (R.Z.); dongzhuoning@buaa.edu.cn (Z.D.); wangyx@buaa.edu.cn (Y.W.)

<sup>2</sup> National Innovation Institute of Defense Technology, Academy of Military Sciences PLA China, Beijing 100071, China; dibin@asee.buaa.edu.cn

\* Correspondence: sunguibinx@buaa.edu.cn; Tel.: +86-10-8231-6849

Received: 23 February 2019; Accepted: 25 April 2019; Published: 28 April 2019



**Abstract:** In this paper, a multi-robot persistent coverage of the region of interest is considered, where persistent coverage and cooperative coverage are addressed simultaneously. Previous works have mainly concentrated on the paths that allow for repeated coverage, but ignored the coverage period requirements of each sub-region. In contrast, this paper presents a combinatorial approach for path planning, which aims to cover mission domains with different task periods while guaranteeing both obstacle avoidance and minimizing the number of robots used. The algorithm first deploys the sensors in the region to satisfy coverage requirements with minimum cost. Then it solves the travelling salesman problem to obtain the frame of the closed path. Finally, the approach partitions the closed path into the fewest segments under the coverage period constraints, and it generates the closed route for each robot on the basis of portioned segments of the closed path. Therefore, each robot can circumnavigate one closed route to cover the different task areas completely and persistently. The numerical simulations show that the proposed approach is feasible to implement the cooperative coverage in consideration of obstacles and coverage period constraints, and the number of robots used is also minimized.

**Keywords:** multi-robot; cooperative coverage; persistent coverage; path planning; coverage period constraints; obstacle avoidance

## 1. Introduction

Research interest for the coverage and coordination of multi-agent has shown an increase in the field of artificial intelligence (AI) and control [1–6]. In particular, coverage control using multiple robots with limited sensing capabilities has received significant attention recently due to its versatility in many applications, such as mapping, patrolling, surveillance, and complete coverage [3–11]. However, it is more difficult to achieve persistent coverage for a group of multiple robots considering obstacle avoidance and coverage period, because mission domains may have differently shaped obstacles, as well as more complicated constraints.

In the current literature, many advanced methods, such as grid-based coverage, cellular decomposition and topological coverage, have been proposed for coverage problems [12,13]. The literature [13] develops two efficient coverage strategies for multiple robots based on boustrophedon cellular decomposition to achieve complete coverage of a known environment. Traditional AI search algorithms, such as A-Star and its variants, have also been applied [14], but cannot adapt to multiple robots. Votion and Cao first develop three improved A-star algorithms to obtain the optimal path, and then they present a new spatially diverse path planning algorithm based on the A-star variants to

address the need for path diversity in multi-agent path planning [15]. Spanning tree coverage [16] is developed for multi-robot area patrolling [17] and surveillance [18], but ignores the sub-regions with different importance. Market-based mechanisms have also been used to assign work to robots, but their applications have been limited [8]. At the other end of the spectrum is the coordination of multi-agent, which includes potential-function based approaches, digital pheromone mechanisms, and particle swarm optimization (PSO) [19–22]. They are effective in dealing with the problem, but often suffer from troubles of local optimum. Another coverage approach includes artificial neural networks (ANNs) to represent control politics [23]. Multi-robot persistent coverage of the convex polygonal region is investigated in [24], where the path consists of the vertices of the scaled barycentric polygons. In [25], an adaptive path planning algorithm is proposed for multiple AUVs cooperative environmental sampling and sensing over an interest region. Atınc, Stipanović, and Voulgaris study a dynamic coverage for multi-agent systems, where the main objective of a group of mobile agents is to explore a given compact region [26]. Franco et al. present a new bounded potential repulsion law to achieve persistent coverage for a team of agents with collision avoidance [19]. The speed controller along the path is further developed in [27] for persistent awareness coverage using mobile sensors. The robots are supposed to be placed on the path uniformly to increase the frequency of revisits in [17]. It is also assumed that the coverage periods of the areas in the region are equal in the literature [27]. Persistent monitoring of given discrete sites under different frequency constraints is considered in [28,29], where the travelling salesman problem (TSP) [28] or the vehicle routing problem with time windows [29] is solved for the routing of the robots.

Most of these methods for multi-robot persistent coverage can be classified into two categories. One class is focussed on the paths to cover the task areas completely and persistently [10,30,31]. They present some new path planning algorithms and implementation for the efficient complete coverage of a known area. However, these methods are only suitable for simple task environments. In addition, they also ignore the sub-regions in the mission domain, which may be more important and need to be re-covered more frequently. The other class is characterized by approaches which are simple and highly scalable to address the problem of multi-robot persistent coverage [32,33]. These approaches can have better performance than a single robot for persistent coverage, but they do not consider using the least number of the robots to cover the areas.

Although the aforementioned methods have promoted the development of coordination algorithm for multi-robot persistent coverage, the number of robots used and coverage period are not taken into account in these cooperative persistent coverage methods. Therefore, this paper presents a new combinatorial approach for cooperative multi-robot path planning, which focuses on the persistent coverage problem with obstacles and different coverage period constraints using the minimum number of robots. The robots are supposed to re-cover the region of interest within every unit of time periodically. The contribution of the combinatorial method with respect to previous works is summarized as three-fold. The first contribution is the development of the proposed path planning based on sensor deployment for cooperative persistent coverage in complex task environments with obstacles. Compared with the traditional coverage planning, the new approach divides path planning issues into sensor deployment problem (SDP) and TSP in order to effectively cope with the planning puzzle caused by environmental obstacles. More precisely, the proposed algorithm introduces the idea of sensor deployment to implement coverage planning in more complex environments, thereby improving the adaptability and robustness of the algorithm to the environment. The second contribution is to consider the coverage period constraints of different sub-regions in the mission domains. The coverage period is utilized to indicate how frequently the region should be re-covered. In the mission area, there may be some sub-regions of different importance depending on the target probability density. Some sub-regions of greater importance are required to be re-covered with smaller periods. In addition, the sensors are deployed to cover the sub-region with the smallest coverage period first, then the sub-region with larger coverage period next, and so on. The third contribution is to optimize the number of robots performing the task for purpose of using a minimum number of

robots to achieve persistent coverage for the mission area. Additionally, the approach can adaptively adjust the number of robots used according to the coverage period constraints of different sub-regions.

The rest of this paper is organized as follows. Section 2 presents the preliminaries to the improved cooperatively coevolving particle swarm optimization (CCPSO2) and modified genetic algorithm (GA). Section 3 describes the problem formulation of multi-robot persistent coverage. In Section 4, a new combinatorial approach for cooperative path planning is developed to achieve the multi-robot persistent coverage. The numerical results of proposed approach are discussed in Section 5. Finally, the discussion and conclusions are made in Sections 6 and 7, respectively.

## 2. Preliminaries

In this paper, we use the CCPSO2 because it is fast enough to find the optimal solution of SDP. In addition, modified GA is utilized to solve TSP because of its inherent parallelism and global search capability.

### 2.1. Improved Cooperatively Coevolving Particle Swarm Optimization

PSO is one of popular swarm intelligence methods. However, the performance of the PSO algorithm deteriorates rapidly as the particle dimension increases. In contrast, the CCPSO2 algorithm decomposes the solution vector into different parts and each part is optimized with a single particle swarm. It reduces the dimension of the solution vector in a single particle swarm and has advantages in solving large-scale optimization problems. Therefore, in consideration of SDP characteristics, CCPSO2 with multiple heuristic rules is introduced to enhance particle diversity and algorithm performance.

In the PSO algorithm, each particle represents a potential solution to the optimization problem. Particles move through the search space to seek the best position. Assume that  $x_i$  and  $y_i$  are the current position and local optimal position of  $i$ th particle respectively. Let  $\hat{y}$  be the global optimal position of particles in the swarm. The update of  $y_i$  and  $\hat{y}$  are determined by:

$$\begin{cases} y_i = x_i, & \text{if } f(x_i) < f(y_i) \\ \hat{y} = \operatorname{argmin}(f(y_i)) \end{cases} \quad (1)$$

where  $f(x_i)$  refers to the fitness value of the particle.

The update of velocity and position are formulated as [34]:

$$\begin{cases} v_{i,d}(t+1) = \omega(t)v_{i,d}(t) + c_1r_1(t)(y_{i,d}(t) - x_{i,d}(t)) + c_2r_2(t)(\hat{y}_{i,d}(t) - x_{i,d}(t)) \\ x_{i,d}(t+1) = x_{i,d}(t) + v_{i,d}(t+1) \end{cases} \quad (2)$$

where  $v_{i,d}(t)$  and  $x_{i,d}(t)$  are the velocity and position of the  $i$ th particle in the  $d$ th dimension respectively;  $\omega(t)$  is the inertial weight;  $c_1$  and  $c_2$  are acceleration constants;  $r_1(t)$  and  $r_2(t)$  are random numbers and satisfy  $r_1(t) \in [0, 1]$ ,  $r_2(t) \in [0, 1]$ .

In order to improve the particle diversity and prevent premature convergence to local optimum, a new update model that uses both Gaussian and Cauchy distributions, as well as ring topology, is proposed in [35].

$$x_{i,d}(t+1) = \begin{cases} y_{i,d}(t) + C(1)|y_{i,d}(t) - y'_{i,d}(t)|, & \text{if } \operatorname{rand} < p \\ y'_{i,d}(t) + N(0, 1)|y_{i,d}(t) - y'_{i,d}(t)|, & \text{otherwise} \end{cases} \quad (3)$$

where  $C(1)$  and  $N(0, 1)$  are the random numbers generated following the Cauchy and Gaussian distributions respectively;  $\operatorname{rand}$  is a random number generated uniformly in the range of  $[0, 1]$ ;  $p$  is a custom parameter for Cauchy sampling to occur;  $y'_i$  denotes a local neighborhood best for the  $i$ th particle.

In this paper, the ring topology is utilized to describe the particles' neighborhood. Each particle is supposed to have an immediate left and right neighbor. Therefore,  $y'_i$  can be defined as:

$$y'_i = \underset{y_i}{\operatorname{argmin}}(f(y_{i-m}), \dots, f(y_i), \dots, f(y_{i+m})) \quad (4)$$

where  $m$  is the neighborhood range. It increases with the number of iterations in this paper. A small  $m$  can improve the particle diversity at the beginning of the optimization process and a larger  $m$  could benefit the convergence at the latter stage of the optimization.

The  $n$ -dimensional solution vectors are decomposed into  $K$  components in the CCPSO2 algorithm. Each component corresponds to a swarm with  $s$  dimensions, where  $s = n/K$ . Suppose that  $P_j.x_i$  and  $P_j.y_i$  are the current position and the local optimal position of the  $i$ th particle of the  $j$ th swarm. Let  $P_j.\hat{y}$  be the global optimal position of the  $j$ th swarm. The current best context vector can be given by  $\hat{y} = (P_1.\hat{y}, P_2.\hat{y}, \dots, P_s.\hat{y})$ .

In order to evaluate the  $i$ th particle of the  $j$ th swarm, substitute  $P_j.x_i$  for  $P_j.\hat{y}$  in  $\hat{y}$ . Hence, define the following combination of particles:

$$b(j, P_j.x_i) = (P_1.\hat{y}, P_2.\hat{y}, \dots, P_{j-1}.\hat{y}, P_j.x_i, P_{j+1}.\hat{y}, \dots, P_s.\hat{y}) \quad (5)$$

The  $P_j.y_i$  can be updated as follows:

$$\begin{cases} P_j.y_i = P_j.x_i, & \text{if } f(b(j, P_j.x_i)) < f(b(j, P_j.y_i)) \\ P_j.\hat{y} = P_j.y_i, & \text{if } f(b(j, P_j.y_i)) < f(b(j, P_j.\hat{y})) \end{cases} \quad (6)$$

Considering the properties of SDP, several heuristic operators for the update of the particles' positions are introduced to improve the convergence speed of the CCPSO2 algorithm.

- Addition. If there are uncovered cells and sensors that have not been deployed in the particle, a sensor is randomly chosen to place near one uncovered cell.

$$\text{if } c_i = 0, \text{ then add } s_{n+1} \text{ at point } c_i \quad (7)$$

where  $c_i$  is a binary vector representing the  $i$ th discretized cell (for details, refer to Section 4.2);  $s_{n+1}$  indicates the new sensor added to the existing  $n$  sensors.

- Movement. If there are uncovered cells in the region and all sensors in the particle have been deployed, a sensor is chosen to move a short distance towards one uncovered cell.

$$\text{if } c_i = 0, \text{ then move } s' \text{ to point } c_i \quad (8)$$

where  $s'$  denotes a certain sensor around the cell  $c_i$ .

- Deletion. If the distance between any deployed sensor and other sensors in the global best is less than the sensing radius of the sensors, the deployed sensor in the particle is deleted.

$$\text{if } \|p(s_i) - p(s_{j,j \neq i})\| < r_s, \text{ then delete } s_i \quad (9)$$

where  $p(s_i)$  is the position of the  $i$ th sensor and  $\|\cdot\|$  refers to the Euclidean distance between  $p(s_i)$  and  $p(s_j)$ .

- Fusion. After all the cells of region have been covered, the fusion operation can be performed. If the distance between any two deployed sensors is less than a certain constant, the two sensors are fused into one sensor with its position at the middle of the two sensors.

$$\text{if } \|p(s_i) - p(s_{j,j \neq i})\| < \tau, \text{ fusion } s_i \text{ and } s_j \quad (10)$$

where  $\tau$  represent a certain constant.

## 2.2. Modified Genetic Algorithm

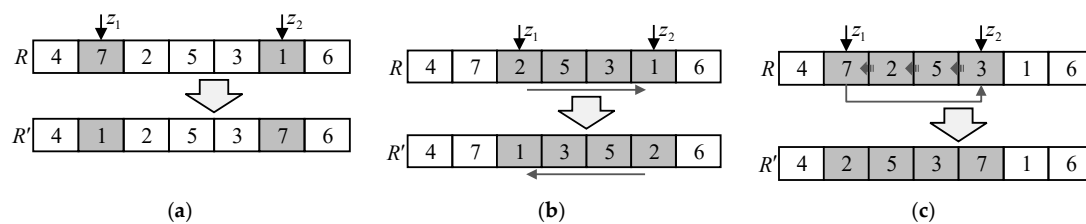
GA is a kind of random search method that has evolved from the evolutionary laws of the biosphere. It could use the probabilistic optimization method to adjust search direction adaptively in the process of solving combinatorial optimization problem. GA is one of the most ideal approaches in solving the TSP because of its inherent parallelism and global search capability. Therefore, the TSP is solved by GA to obtain the closed path in this paper.

Assume that there are  $n$  cities and each city is represented by an integer from 1 to  $n$ . In this way, chromosome  $R_k$  can be described as  $R_k = \{g_1, g_2, g_3, \dots, g_n\}$ ,  $g_i \in [1, n]$ , where  $g_i$  is the  $i$ th gene and also represents a city.  $R_k$  is a chromosome and refers to the  $k$ th feasible path as well. It is a gene sequence consisting of  $n$  genes and each gene of the chromosome is different from each other. For example, there are 5 cities in the TSP, then  $\{1, 3, 4, 2, 5\}$  is a legitimate chromosome and a possible optimal solution. It means that the salesman visits cities 1, 3, 4, 2, and 5 in order. Suppose that  $X(j)$  represents the  $j$ th population. It can be expressed as  $X(j) = \{x_1^j, x_2^j, x_3^j, \dots, x_m^j\}$ , where  $m$  is the size of the population;  $x_i^j$  refers to the  $i$ th individual of the  $j$ th population. In this paper, a random function is used to generate an initialization population  $X(0) = \{x_1^0, x_2^0, x_3^0, \dots, x_m^0\}$ .

There are three basic operations in the standard GA, that is, selection operation, crossover operation, and mutation operation. Considering the characteristics of the TSP, this paper uses following four operations: selection operation, mutation operation, evolutionary reversal operation, and slide operation.

The selection operation is to generate a new population with higher fitness value. It is selected by the selection probability from the current population. The main objective of selection operation is to inherit the high-quality genes to the next generation while ensuring fast global convergence. In this paper, the individual with the maximum fitness value  $f(x_i^j)$  is utilized to generate a new population as the elite individual.

Mutation operation is a very important operator of the GA. In this paper, the mutation operation adopts a strategy of randomly exchanging two genes of one chromosome. As depicted in Figure 1a, there is a chromosome  $R$  consisting of 7 genes and randomly generate two gene positions  $z_1 = 2$  and  $z_2 = 6$ . Then exchange the genes at these two positions to obtain the new chromosome  $R'$ .



**Figure 1.** Schematic diagrams of three operations. (a) Mutation operation using randomly exchanging two genes of one chromosome; (b) Evolutionary reversal operation through re-sorting genes between two random gene positions; (c) Slide operation via rotating one gene position left between two random gene positions.

In order to improve the local search ability of the GA, the evolutionary reversal operation is introduced after the selection operation and mutation operation. The evolution refers to the unidirectionality of reversal operator, that is, after the reversal operation, the individual will perform the operation if it becomes better, otherwise the reversal is invalid. The method is to randomly generate two random numbers  $z_1$  and  $z_2$ , and then the genes between  $z_1$  and  $z_2$  are re-sorted in reverse order. The process of the evolutionary reversal operation is depicted in Figure 1b. Assume that there are 7 genes in one chromosome and randomly generate two gene positions  $z_1 = 3$  and  $z_2 = 6$ . Then re-sort the genes between  $z_1$  and  $z_2$  in reverse order.

Slide operation can greatly inherit the advantages of the parent individual and it can also prevent the algorithm from falling into local optimum. Figure 1c shows the process of slide operation. Firstly,

two gene positions  $z_1 = 2$  and  $z_2 = 5$  are randomly generated. Then rotate one gene position left between  $z_1$  and  $z_2$ .

### 3. Problem Formulation

#### 3.1. Basic Assumptions

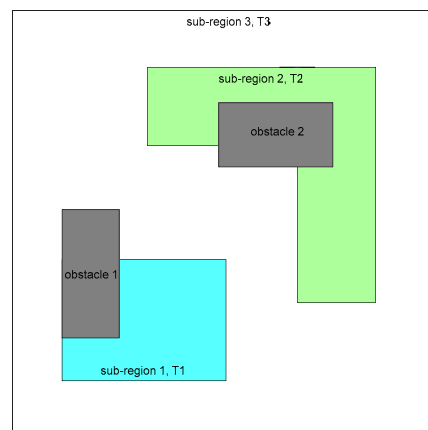
In this paper, a new combinatorial approach is proposed for multi-robot persistent coverage. The following conditions are assumed to describe the cooperative persistent coverage.

- Each robot is supposed to be homogeneous.
- The detection zone of a sensor is simplified to a circle, and the influence of robot motion on sensing range is not considered.
- The center of the sensor detection zone, namely the center of detection circle, is considered as the position of the sensor.
- The position of a sensor is regarded as the robot observation point, that is, the robot waypoint. When the robot is at the observation point, the area within the coverage of the sensor can be detected.
- The probability density of target in each sub-region is known by the priori information, and there are different coverage period requirements for each sub-region.
- The total velocity of each robot is set to the constant value.

#### 3.2. Problem Statement

Suppose that there is a group of robots performing persistent coverage task over the given region. The sensing radius of each robot is denoted as  $r_s$ . The region of interest  $R_{OI}$  consists of the feasible region  $R_f$  and the obstacle region  $R_o$ , which satisfy  $R_f \cup R_o = R_{OI}$  and  $R_f \cap R_o = \emptyset$ . The obstacles cannot be reached and would also block the sight of the sensors.

The robots are tasked to re-cover the feasible region within every  $T$  units of time in order to update their observations continually. The coverage period  $T$  is adopted to indicate how frequently the region should be re-covered. Some sub-regions of more importance in the feasible region may be required to be re-covered with smaller periods. It is assumed that the coverage periods of the feasible region and the sub-regions are known according to the prior information. As depicted in Figure 2, there are three sub-regions (sub-regions 1, 2, and 3) and two obstacles (obstacles 1 and 2) in the given region. Sub-regions 1 and 2 are more important and required to be re-covered with smaller coverage periods  $T_1$  and  $T_2$ , respectively. The rest of the feasible region is denoted as the sub-region 3 with coverage period  $T_3$ .



**Figure 2.** Sub-regions and obstacles in the task area. The cyan area is sub-region 1. The green area is sub-region 2. The white area is sub-region 3. Gray areas refer to obstacles.

Given the capabilities of the robots and the coverage period of each sub-region, the persistent coverage problem is translated into how to plan the robots' paths with the coverage periods satisfied, using a minimum number of robots. There are two correlative questions in the persistent coverage problem: (1) How to plan a closed path that can effectively avoid geometrical obstacles; (2) How many robots are required at least to satisfy the coverage period constraints?

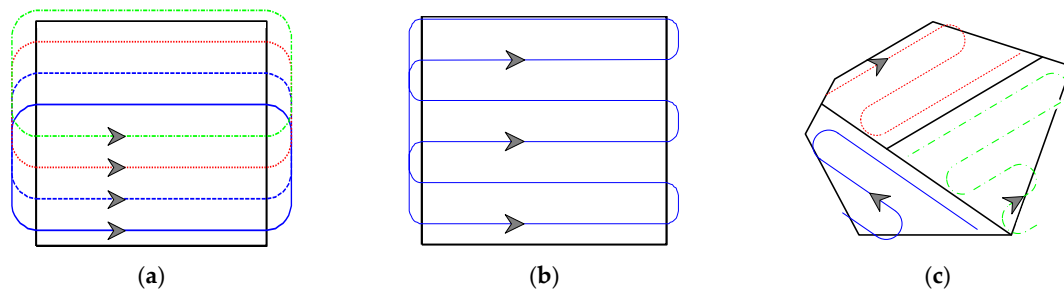
In this paper, we first consider the path planning for complete coverage in the region. Due to the complexity caused by the obstacles, the path planning problem is decomposed into SDP and TSP. Then, partition the path into several segments considering the sub-regions with different task importance and the robots with optimum quantity. In order to solve the problem efficiently, we divide the solution strategy into three steps. Firstly, the feasible region  $R_f$  is covered completely using the virtual sensors with sensing radius  $r_v$ . The purpose of SDP research is to deploy the sensors in the region to satisfy coverage requirements and ensure minimum cost [36–42]. Sensor deployment in the feasible region can be performed in several stages, according to the coverage periods of sub-regions from small to large. The sub-region with the smallest coverage period is preferentially deployed. Then the sub-region with the larger coverage period is deployed until all the sub-regions are completely covered. Secondly, taking the positions of the deployed sensors as the waypoints, the TSP is solved to obtain the closed path which connects all the sensors. The feasible region can be covered completely if a robot circumnavigates the closed path once. Thus, we call the closed path 'the frame'. At last, in consideration of the coverage period constraints and the optimum quantity of robots, the frame is partitioned into several segments with the aim of minimizing the number of the segments. On the basis of each partitioned segment, the closed route is generated for each robot. Furthermore, each robot is assigned one closed route to circumnavigate in order to achieve persistent coverage of the feasible region.

The purpose of this paper is to develop a new combinatorial approach for multi-robot persistent coverage, which aims to cover mission domains with different task periods while guaranteeing both the obstacle avoidance and the optimum number of robots. In this paper, the improved CCPSO2, modified GA, and PSO are applied to design the combinatorial algorithm.

## 4. Combinatorial Approach for Multi-Robot Persistent Coverage

### 4.1. Traditional Cooperative Persistent Coverage

Multi-robot cooperative coverage can effectively improve mission efficiency in the task of persistent coverage. In a relatively simple environment, in order to ensure complete coverage of the area, the geometry-based approach is often used to search the area. Figure 3a illustrates an example of the parallel scanning coverage. After determining the number of robots, the routes of each robot are generated according to the geometric rules, in consideration of sensor's detection width, turning radius, and entering direction. Under this scan strategy, the increase in the number of robots widens the overall scan radius, which reduces the search period and improves coverage efficiency. Sequential scanning coverage is shown in Figure 3b. The method first obtains the path of a single robot persistent coverage. Then each robot moves on the planned path sequentially. Robots are evenly distributed on the route at a certain interval, which reduces the coverage period and achieves a better coverage. The geometry-based approach is a common strategy for multi-robot coverage. However, this approach is only suitable for the regular task area with no obstacles. It cannot effectively address persistent coverage issues in complex task environments with target probabilities and obstacles.



**Figure 3.** Three schematic diagrams of persistent coverage based on different methods. (a) An example of the parallel scanning coverage; (b) An example of the sequential scanning coverage; (c) An example of the area-decomposition based coverage.

Area decomposition technique is another strategy for multi-robot persistent coverage. The approach first divides the mission area into several sub-regions, and then the robots are tasked to re-cover the respective sub-regions. The literature [43] shows a cooperative search using multiple unmanned air vehicles (UAV). The algorithm divides mission area into several sub-regions in terms of the UAVs' initial positions and the percent of search area of each UAV. Then each UAV searches respective sub-region and selects the appropriate direction to reduce the number of turns. The planning result of Ref. [43] is illustrated in Figure 3c. The advantage of area decomposition is that the straight part of the coverage route is longer, which can reduce the number of turns and boost the coverage efficiency. However, it is necessary to consider the influence of obstacles on the area division when the mission environment becomes more complicated. At this time, it is extremely difficult to divide the region according to geometric rules. In addition, methods based on area decomposition neglect the sub-regions in the mission area, which may have different importance and need to be re-covered with different coverage periods.

In summary, current methods for multi-robot persistent coverage mainly have the following defects:

**Problem 1.** They have failed to address cooperative persistent coverage effectively in complex mission environments with obstacles.

**Problem 2.** Previous works neglect the coverage period of each sub-region in the mission area. Some sub-regions of more importance may be re-covered with different coverage periods.

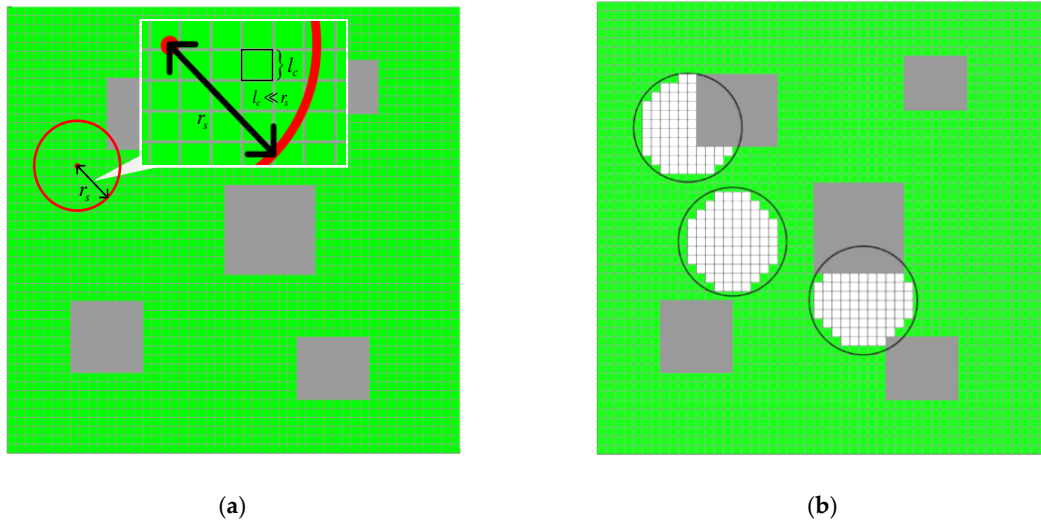
**Problem 3.** Current methods do not consider how to divide the task region and assign the robots, in order to improve coverage efficiency.

Based on the above facts, the combinatorial method proposed in this paper focuses on the multi-robot persistent coverage with obstacles and different coverage period constraints, using a minimum number of robots. Cooperative persistent coverage can be divided into the following three steps to deal with: (1) sensor deployment in the task area; (2) path planning based on the TSP; (3) partition of closed path considering coverage period.

#### 4.2. Sensor Deployment in The Task Area

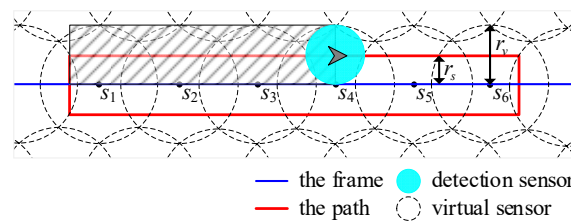
The coverage region is a bordered area. Once the mission area is given, it can be divided into many rectangular cells whose side length  $l_c$  is much smaller than the sensing radius  $r_s$ . Let  $C_f = \{c_1, c_1, \dots, c_m\}$  be the set of cells in the feasible region  $R_f$ . As shown in Figure 4a, the task region is divided into rectangular cells of  $50 \times 50$  and the side length of each cell satisfies  $l_c \ll r_s$ .





**Figure 4.** (a) Discretization of coverage area. The task region is divided into rectangular cells of  $50 \times 50$ . Grey rectangles refer to the obstacles. (b) The coverage of the sensors in the presence of obstacles. White cells indicate that they have been covered by the virtual sensors. Green cells represent that they are uncovered by any virtual sensor.

The feasible region is covered completely using the virtual sensors with sensing radius  $r_v$ , where  $r_v = 2r_s$ . Suppose that  $S = \{s_1, s_2, \dots, s_n\}$  is the set of the sensors. Figure 5 illustrates an example of sensor deployment. The task area is covered by the virtual sensors. The blue line is a segment of the frame, namely the closed path connecting all the waypoints in the mission domains. The red closed route is generated on the basis of portioned segment and robot's sensing range. The robot carrying detection sensor, with sensing radius  $r_s$ , is tasked to track the red route. As depicted in Figure 9, one sensor covers two-part routes of the red closed route. Therefore, the radius of virtual sensor is twice the sensing radius of robot, that is  $r_v = 2r_s$ .



**Figure 5.** An example of sensor deployment. The radius of each robot is  $r_s$  and the radius of the virtual sensor is  $r_v$ . The shaded area represents the field that robots have scanned.

The cell  $c_i$  can be covered by the sensor  $s_j$  if all the four vertices of the cell  $c_i$  are within the sensing range of the sensor  $s_j$  and are not blocked by any obstacle. The coverage situation of the sensors is shown in Figure 4b. Let  $e_i = (p(s_i), g_i)$  denote the element which embodies the basic deployment information of the sensor  $s_i$ , where  $p(s_i)$  is the position of the sensor  $s_i$ , and  $g_i$  is the validity flag. Assume that  $g_i = 1$  if the sensor  $s_i$  is deployed, otherwise  $g_i = 0$ . Thus, the deployment vector  $[e_1, e_2, e_3, \dots, e_k]$  could represent a deployment situation of the sensors, namely a feasible solution to SDP, where  $k$  is the maximum number of the available sensors.

Every point in the feasible region is supposed to be revisited periodically. Therefore, the feasible region should be covered completely in SDP. Fewer sensors and less overlap of cells could cause a shorter path, and hence improve the efficiency of persistent coverage task. Taking the objectives of the sensor deployment into account, the fitness function of the sensor deployment is defined as:

$$f_s(S) = \lambda_1 \times n_b + \lambda_2 \times n_u + \lambda_3 \times n_s + \lambda_4 \times n_o \tag{11}$$

where  $S$  indicates a feasible solution to SDP;  $n_b$ ,  $n_u$ ,  $n_s$ , and  $n_o$  are the number of the sensors deployed in the obstacles, the number of the uncovered cells, the number of sensors deployed, and the number of the overlapped cells, respectively;  $\lambda_1$ ,  $\lambda_2$ ,  $\lambda_3$ , and  $\lambda_4$  represent the corresponding weighted factors, which satisfy  $\lambda_1 \gg \lambda_2 \gg \lambda_3 > \lambda_4 > 0$ .

The fitness function is minimized based on the improved CCPSO2 algorithm to obtain the optimal deployment vector. The CCPSO2 is adopted due to its ability for solving large-scale optimization problems. Considering the properties of SDP, several heuristic operators are introduced to promote the performance of the normal CCPSO2. As depicted in Figure 6, Figure 6a shows the sensor deployment using improved CCPSO2 with several heuristic rules and Figure 6b illustrated the deployment situation using standard CCPSO2 with no heuristic rules. There are 99 sensors used in Figure 6a while a total of 103 sensors are used in Figure 6b. According to the cost curves for two cases in Figure 6c, it can be found that the introduction of heuristic rules can effectively improve the convergence speed of the CCPSO2. In addition, the heuristic operators can also improve the diversity of the particles, and hence improve the performance of the algorithm in solving SDP. Submitting (3) to (6), the pseudo-code for sensor deployment is listed in **Algorithm 1**.

**Algorithm 1.** Pseudo-code of improved cooperatively coevolving particle swarm optimization (CCPSO2) for sensor deployment.

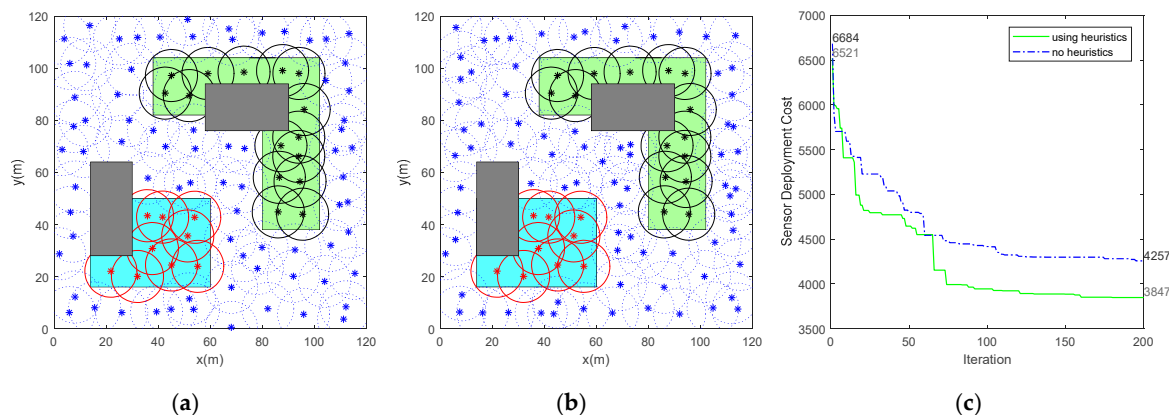
---

**Algorithm 1:** Improved CCPSO2 for sensor deployment.

---

|  |  |
|--|--|
| <pre> 01: // Initialization: 02: Set task parameters. 03: Set parameters of improved CCPSO2. 04: Randomly initialize <math>K</math> swarms, each     with <math>s</math> sensors. 05: // Main loop: 06: Repeat each iteration 07: if <math>f_s(\hat{y})</math> has not been improved,     Choose value <math>s</math> from a predefined     set randomly, contrast <math>K</math> (<math>K = n/s</math>). 08: end 09: // Update optimal position 10: for each swarm <math>j</math> 11:   for each particle <math>i</math> 12:     <math>P_j.y_i</math> and <math>P_j.\hat{y}_i</math> are updated as (6). 13:   end 14:   for each particle <math>i</math> 15:     <math>P_j.y'_i</math> is updated as (5). 16:   end 17:   Compute fitness value of <math>b(j, P_j.x_i)</math> 18:   and <math>y'_i</math>. 19:   if <math>f_s(b(j, P_j.x_i)) &lt; f_s(\hat{y}_i)</math> 20:     <math>b(j, P_j.x_i) = P_j.\hat{y}_i</math> 21:   end 22: end 23: // Update particles in each swarm 24: for each swarm <math>j</math> 25:   for each particle <math>i</math> </pre> | <pre> 26:     // Update particle by heuristic rules 27:     if <math>rand &lt; p</math> 28:       Remove the sensor with closer     distance via <b>Deletion</b> operator. 29:       if the task area is covered     completely 30:         Fuse and move sensor by <b>Fusion</b>     and <b>Movement</b> operators. 31:       end 32:     else 33:       Add and move sensor by <b>Addition</b> and <b>Movement</b> operators 34:     end 35:   end 36: // Update particle by CCPSO2 rules 37: else 38:   if <math>rand &lt; q</math> 39:     The <math>i</math>th particle is updated as (2). 40:   end 41: else 42:     The <math>i</math>th particle is updated as (3). 43:   end 44: end 45: end 46: end 47: end 48: // Results: 49: Find the optimal solution as the     sensor deployment situation. </pre> |
|--|--|

---



**Figure 6.** (a) Sensor deployment using improved cooperatively coevolving particle swarm optimization (CCPSO2) with heuristic rules. (b) Sensor deployment using normal CCPSO2 with no heuristic rules. (c) Sensor deployment cost for two situations.

In this paper, the sensors are deployed to cover the sub-region with the smallest coverage period first, then the sub-region with larger coverage period, and so on. As shown in Figure 2, there are three sub-regions in the task area. The task coverage period satisfies  $T_1 > T_2 > T_3$ . This means that sub-region 1 is more important than sub-region 2, and sub-region 2 is more important than sub-region 3. Therefore, the order of sub-regions to be covered by the sensors is [sub-region 1, sub-region 2, and sub-region 3]. After determining the coverage order, each sub-region is sequentially covered according to **Algorithm 1**. The sub-regions that have been covered by the sensors in the former phases are seen as obstacles when deploying sensors over other sub-regions in the latter phase. It should be noted that the sensor deployment of each sub-region is optimized independently. In this way, it can reduce the complexity of the algorithm and speed up the convergence process, as well as facilitate partition of closed path in subsequent work.

### 4.3. Path Planning Based on Travelling Salesman Problem

Taking positions of deployed sensors as the waypoints, TSP is solved to obtain the frame of the path for persistent coverage. The feasible region can be covered completely if a robot circumnavigates the closed path once.

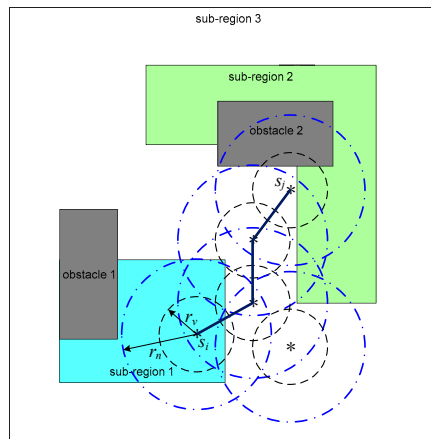
Let  $r_n$  be the neighboring range and  $p(s_i)$  be the position of the sensor  $s_i$ . Taking waypoints as vertices, we define the proximity graph  $G = (V, E)$ , where  $V$  and  $E$  denote the vertices and edges.  $E$  is defined as  $E = \{(s_i, s_j) \in V \times V : \|p(s_i) - p(s_j)\| \leq r_n, s_i \neq s_j\}$ , where  $\|p(s_i) - p(s_j)\|$  is the Euclidean distance between the vertices  $s_i$  and  $s_j$ . A path that connects the vertices  $s_1$  and  $s_n$  is defined as a sequence of the distinct vertices  $sq_i(s_1, s_n) = [s_1, s_2, \dots, s_i, \dots, s_n]$ , which satisfies  $(s_i, s_{i+1}) \in E, 1 \leq i \leq n - 1$ . The graph  $G$  is connected if there is a path between any two distinct vertices. If the feasible region is covered by the deployed sensors completely and the neighboring range  $r_n$  is no smaller than twice the virtual sensor radius  $r_v$ , that is  $r_n \geq 2r_v$ , the proximity graph  $G$  is connected [44]. A path that connects the sensors  $s_i$  and  $s_j$  is illustrated in Figure 7, where the neighboring range  $r_n = 2r_s$ .

Let  $sr(s_i)$  denote the sub-region in which the sensor  $s_i$  is deployed. The length of the path  $sq_i(s_1, s_n)$  is defined as:

$$l(sq_i(s_1, s_n)) = \sum_{i=1}^{n-1} (\|p(s_i) - p(s_{i+1})\|) + \sum_{i=1}^{n-1} \alpha I(sr(s_i) \neq sr(s_{i+1})) \tag{12}$$

where  $\|p(s_i) - p(s_{i+1})\|$  is the Euclidean distance between the vertices  $s_i$  and  $s_{i+1}$ ;  $\alpha > 0$  is a weighted constant;  $I(\cdot)$  denotes the indicator function, which satisfies:

$$I(sr(s_i) \neq sr(s_{i+1})) = \begin{cases} 1, sr(s_i) \neq sr(s_{i+1}) \\ 0, sr(s_i) = sr(s_{i+1}) \end{cases} \tag{13}$$



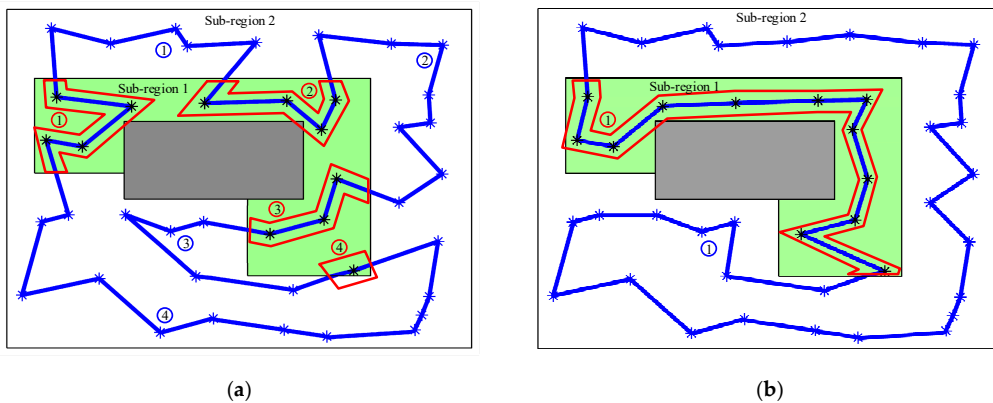
**Figure 7.** A path that connects the sensors  $s_i$  and  $s_j$ . Black circle is the sensing range of virtual sensor. Black asterisk refers to the position of the sensor. Blue circle denotes the neighboring radius of the virtual sensor.

In order to reduce the connections between different sub-regions in the frame of the path, instead of the Euclidean distance, the distance between two sensors  $s_1$  and  $s_n$  is defined as:

$$d(s_1, s_n) = \min_{sq_i(s_1, s_n)} (l(sq_i(s_1, s_n))), sq_i(s_1, s_n) \in SQ(s_1, s_n) \tag{14}$$

where  $SQ(s_1, s_n)$  is the set of the paths that connect the sensors  $s_1$  and  $s_n$ .

In the task of cooperative persistent coverage, the path frame needs to be divided into several segments. What is more, fewer connections among different sub-regions would be beneficial to reduce the number of segments, i.e., the number of the robots. Therefore, in order to facilitate path segmentation in subsequent work, it is necessary to reduce connections among different sub-regions as much as possible. The frames obtained based on different distance definitions are presented in Figure 8. The Euclidean distance is used for path planning in Case 1. It can be seen that the frame spanned eight times between sub-region 1 and sub-region 2. In addition, the path frame is divided into eight segments by the boundaries. Among them, there are four segments in sub-region 1, and four segments in sub-region 2. While in Case 2, the distance between two sensors is defined by (14) for path planning. It is clear that the frame only crosses twice between sub-region 1 and sub-region 2. In the same time, the path is divided into two sections, each sub-region with a section. Obviously, the frame in Case 2, effectively reduces the connections in different sub-regions, and the planning result is more suitable for the path segmentation.



**Figure 8.** Frame generated based on different distance definitions. Grey rectangles refer to obstacles. Green area indicates sub-region 1. White area is sub-region 2. (a) Using Euclidean distance. (b) Using distance defined by (14).

The shortest path that connects any two waypoints as well as its length is obtained based on the A-star algorithm. The evaluation function is in the form of:

$$f_a(M) = g(M) + h(M) \quad (15)$$

where  $g(M)$  refers to the actual cost function of the path that has passed, from the start point to the current point  $M$ ;  $h(M)$  indicates the heuristic function of the remaining path, from the current point  $M$  to the target point.  $g(M)$  can be formulated as:

$$g(M) = \tau_1 \cdot f(M-1) + \tau_2 \cdot D_{cur} + \tau_3 \cdot P_{obs} \quad (16)$$

where  $f(M-1)$  is the cost value from the start point to the  $M-1$ th point;  $D_{cur}$  denotes the distance between previous point  $M-1$  and current point  $M$ ;  $\tau_1$ ,  $\tau_2$ , and  $\tau_3$  are the weighting coefficients of each item. The heuristic function can be estimated by the following expression:

$$h(M) = \tau_4 \cdot D_{res} \quad (17)$$

where  $D_{res}$  represents the distance from current point  $M$  to the target point;  $\tau_4$  is the weighting coefficient.

In this paper, modified GA is used to solve TSP on account of its inherent parallelism and global search capability. The fitness function can be defined by:

$$f_t(sq_i(s_1, s_n)) = \frac{1}{l(sq_i(s_1, s_n)) + D(s_n, s_1)} \quad (18)$$

where  $sq_i(s_1, s_n)$  refers to the  $i$ th feasible path;  $D(s_n, s_1)$  indicates the distance between the city  $s_n$  and  $s_1$ , which satisfies:

$$D(s_n, s_1) = \|p(s_n) - p(s_1)\| + \alpha I(sr(s_n) \neq sr(s_1)) \quad (19)$$

where  $\|p(s_n) - p(s_1)\|$  is the Euclidean distance between the city  $s_n$  and  $s_1$ ;  $\alpha > 0$  is a weighted constant;  $(\cdot)$  denotes the indicator function. Then, the TSP can be formulated by the modified GA as follows.

**Algorithm 2.** Pseudo-code of modified genetic algorithm (GA) to solve the travelling salesman problem (TSP).

---

**Algorithm 2:** Modified GA to solve TSP.

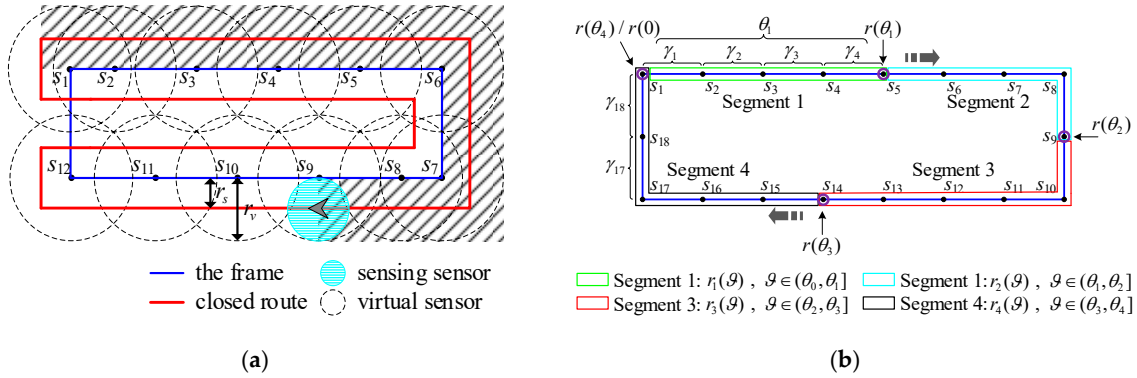
---

|   |   |
|---|---|
| 01: // <b>Initialization:</b>   | 13: <b>switch</b> $k$   |
| 02: Obtain sensor locations from the result of <b>Algorithm 1.</b>                    | 14: <b>case 1</b> No operation on $tem\_pop(1)$                                   |
| 03: Set parameters of modified genetic algorithm.                                     | 15: <b>case 2</b> Perform <b>Mutation operation</b> on $tem\_pop(2)$              |
| 04: Randomly initialize $4 \times K$ populations.                                     | 16: <b>case 3</b> Perform <b>Evolutionary reversal operation</b> on $tem\_pop(3)$ |
| 05: // <b>Main loop:</b>  | 17: <b>case 4</b> Perform <b>Slide operation</b> on $tem\_pop(4)$                 |
| 06: <b>Repeat</b> each generation   | 18: <b>end</b>  |
| 07: Calculate individual fitness values for each population.                          | 19: <b>end</b>  |
| 08: <b>Selection operation:</b> Search the individual with the highest fitness value. | 20: Use $tem\_pop$ to generate $4 \times K$ new populations.                      |
| 09: Update optimal individual $opt\_pop$ .  | 21: <b>end</b>  |
| 10: // Generate new populations   | 22: // <b>Results:</b>  |
| 11: <b>for</b> $k = 1 : 4$  | 23: Get the best path found by the algorithm.                                     |
| 12: $tem\_pop(k) = opt\_pop$  |   |

---

4.4. Partition of Closed Path Considering Coverage Period

Given the frame obtained by solving the travelling salesman problem, the path for complete coverage can be obtained by circumnavigating the frame, as depicted in Figure 9a. The blue line is the frame of the closed path, which connects all the waypoints in the mission area. The red closed route is generated on the basis of the blue frame and robot’s sensing range. The robot carrying detection sensor with sensing radius  $r_s$  circumnavigates the red route to achieve full coverage of the current region.



**Figure 9.** (a) The schematic diagram of the closed route for complete coverage generated by the frame. The radius of each robot is  $r_s$  and the radius of virtual sensor is  $r_v$ , which satisfies  $r_v = 2r_s$ . The shaded area is the field that robots have scanned. (b) An example of path partition. Segment 1 is inside the green box. Segment 2 is inside the cyan box. Segment 3 is inside the red box. Segment 4 is inside the black box. Termination point of each segment is inside the purple circle.

In order to satisfy the coverage period constraints, multiple robots may be required. Taking the coverage periods of the sub-regions into account, the frame is partitioned into several segments with the aim of minimizing the number of the segments. On the basis of each partitioned segment, the closed route is generated for each robot. Afterwards, each robot is assigned one closed route to circumnavigate in order to achieve persistent coverage of the feasible region. Assume that  $\gamma_i$  refers to the normalized length between the sensor  $s_i$  and sensor  $s_{i+1}$ . It can be determined by the following expression:

$$\gamma_i = \frac{d(s_i, s_{i+1})}{D_{sum}}, \gamma_i \in [0, 1] \tag{20}$$

where  $d(s_i, s_{i+1})$  is the actual length of the  $i$ th segment;  $D_{sum}$  represents the length of the frame of the closed path. Total path length  $D_{sum}$  can be given by:

$$D_{sum} = \sum_{i=1}^n d(s_i, s_{i+1}) \tag{21}$$

where  $n$  is the number of sensors deployed; the sensor  $s_{n+1}$  refers to the sensor  $s_1$ , which satisfies  $d(s_n, s_{n+1}) = d(s_n, s_1)$ .

Let the frame of the closed path be denoted as a closed curve. The curve equation can be expressed as  $r(\vartheta) : [0, 1] \rightarrow \mathbb{R}^2$ ,  $r(0) = r(1)$ , where  $\vartheta$  is the normalized length of the curve.  $r(0)$  denotes the initial point of the curve, which is the location of sensor  $s_1$ . Simultaneously,  $r(0)$  is also the initial point of curve segmentation. Similarly,  $r(1)$  is the end point of the curve. Since the curve is closed, there is  $r(0) = r(1)$ . It should be noted that the curve equation  $r(\vartheta)$  is a function of the normalized length  $\vartheta$ .

Suppose that the curve is partitioned into  $m$  segments by  $[\theta_1, \theta_2, \dots, \theta_m]$ , where  $\theta_1 < \dots < \theta_m$ . Param  $\theta_i$  represents the normalized length between the initial point  $r(0)$  and the termination point  $r(\theta_i)$  of the  $i$ th segment, which is defined as:

$$\theta_i = \sum_{k=1}^K \gamma_k, \quad 1 \leq k \leq K, \quad 1 < K \leq n \quad (22)$$

where  $\gamma_k$  is the normalized length between the sensor  $s_k$  and sensor  $s_{k+1}$ ;  $K$  refers to the serial number of the last sensor contained in the  $i$ th segment;  $n$  is the number of sensors.

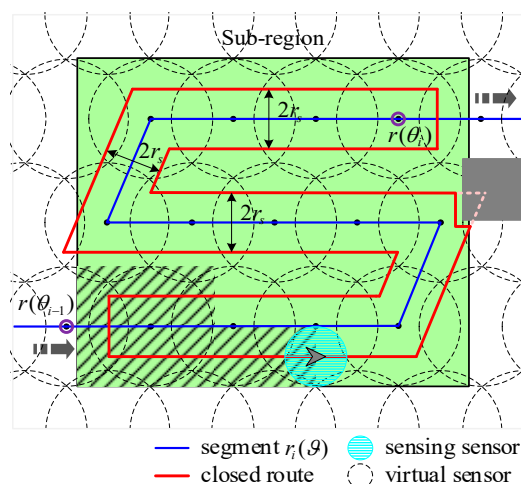
Let  $r_i(\vartheta)$  be the  $i$ th segment, which can be described as  $r_i(\vartheta) : (\theta_{i-1}, \theta_i] \rightarrow \mathbb{R}^2$ ,  $1 \leq i \leq m$ , where  $\theta_i$  is the normalized length between the  $r(0)$  and the termination point  $r(\theta_i)$  of the current segment;  $r(\theta_m)$  refers to the end point  $r(1)$  of the curve;  $r(\theta_0)$  represents the initial point  $r(0)$  of the curve. It should be noted that the following formula is established  $r(\theta_m) = r(1) = r(0) = r(\theta_0)$ . Additionally,  $r(\theta_i)$  and  $r_i(\theta_i)$  are the same point, which satisfies  $r_i(\theta_i) = r(\theta_i) = p(s_K)$ , where  $p(s_K)$  denotes the position of the last sensor in the  $i$ th segment.

Therefore, the problem of frame partition is turned into finding the optimal termination point  $r(\theta_i)$  for each segment according to the coverage period constrains. The point  $r(\theta_i)$  also represents the best location  $p(s_K)$  of the last sensor  $s_K$  contained in each segment. Figure 9b illustrates an example of path partition. The closed path is divided into four segments:  $r_1(\vartheta)$ ,  $r_2(\vartheta)$ ,  $r_3(\vartheta)$ , and  $r_4(\vartheta)$ . There are eighteen sensors in the closed path. Among them, segment 1 contains four sensors from  $s_2$  to  $s_5$ ; segment 2 involves four sensors from  $s_6$  to  $s_9$ ; segment 3 contains five sensors from  $s_{10}$  to  $s_{14}$ ; segment 4 includes five sensors from  $s_{15}$  to  $s_{18}$ .

Let  $T^c(s_j)$  be the coverage period of the sub-region where the sensor  $s_j$  is deployed. Let  $S_i$  be the set of the sensors which are deployed on the segment  $r_i(\vartheta)$ , where  $\vartheta$  satisfies  $\vartheta \in (\theta_i, \theta_{i+1}]$ . The coverage period of the  $i$ th segment  $r_i(\vartheta)$  can be written in the equivalent form as follows:

$$T_i^c = \min_{s_j \in S_i} (T^c(s_j)) \quad (23)$$

Assume  $R_c(r_i(\vartheta))$  is the closed curve generated by the segment  $r_i(\vartheta)$  based on computational geometry.  $R_c(r_i(\vartheta))$  also represents the closed route assigned to each robot. As shown in Figure 10, red line refers to the closed route generated by the blue segment  $r_i(\vartheta)$ . The width of the closed route is twice the sensing radius of each robot, that is  $2r_s$ . When the obstacle blocks the closed route, the method will re-adjust the closed route according to the threat range of the obstacle.



**Figure 10.** An example of the closed route generated by the segment. Green area indicates sub-region. Grey rectangle is obstacle. The shaded area is the field that robots have scanned. The termination point of each segment is inside the purple circle.

The purpose of the frame partition is to minimize the number of robots performing tasks, while meeting coverage period constraints of different sub-regions. Consequently, the objective of the frame partition problem can be described as:

$$\min m \quad (24)$$

subject to:

$$T_i^a < T_i^c, \forall i \in \{1, 2, \dots, m\} \quad (25)$$

where  $T_i^a$  is the time required for a robot to circumnavigate the closed route  $R_c(r_i(\vartheta))$  once.  $T_i^a$  can be obtained by the length of the closed path and robot's velocity. It is given in the following expression as:

$$T_i^a = \frac{L(R_c(r_i(\vartheta)))}{V}, \vartheta \in (\theta_{i-1}, \theta_i] \quad (26)$$

where  $L(R_c(r_i(\vartheta)))$  denotes the length of the closed route  $R_c(r_i(\vartheta))$ ;  $V$  is the velocity of robot. It is assumed that the robots move at a constant speed for simplicity.

The frame partition problem is solved based on the PSO due to its flexibility and global optimization capability [45]. The update of velocity and position are determined by:

$$\begin{cases} v_i(t+1) = \omega(t)v_i(t) + c_1r_1(t)(y_i(t) - x_i(t)) + c_2r_2(t)(\hat{y}(t) - x_i(t)) \\ x_i(t+1) = x_i(t) + v_i(t+1) \end{cases} \quad (27)$$

where  $v_i(t)$  and  $x_i(t)$  represent the velocity and position of the  $i$ th particle respectively;  $y_i(t)$  is the local best position for the  $i$ th particle;  $\hat{y}(t)$  denotes the global optimal position of particles in the swarm;  $\omega(t)$  is the inertial weight;  $c_1$  and  $c_2$  are acceleration constants;  $r_1(t)$  and  $r_2(t)$  refer to the random numbers, which satisfy  $r_1(t) \in [0, 1]$ ,  $r_2(t) \in [0, 1]$ .

In the path partition algorithm, actual coverage time  $T_i^a$ , coverage period constraint  $T_i^c$  and the number of segments  $N_{seg}$  are the main considerations for the frame partition of closed path. Thus, the fitness function can be defined as:

$$f_p(\chi) = \xi_1 \times T_{dif} + \xi_2 \times N_{seg} + \xi_3 \times T_{err} \quad (28)$$

where  $\chi$  is a feasible solution to the frame partition;  $T_{dif}$  indicates the time difference between the actual coverage time  $T_i^a$  and coverage period constraint  $T_i^c$ , which satisfies  $T_{con} = T_i^a - T_i^c + T_m$ ;  $T_{err}$  refers to the time offset between  $T_i^a$  and  $T_i^c$ , which satisfies  $T_{err} = T_i^c - T_i^a - T_m$ ;  $\xi_1$ ,  $\xi_2$ , and  $\xi_3$  are the weighting coefficients of each item. Constant  $T_m$  represents the time margin, which can adjust the time offset to meet the coverage period requirements. Feasible solution  $\chi$  can be expressed as  $\chi = \{r(\theta_1), r(\theta_2), \dots, r(\theta_m)\}$ , where  $r(\theta_i)$  is the termination point of the  $i$ th segment;  $m$  is the number of segments. In the path partition algorithm, the PSO optimizes a set of locations, which are the positions of the last sensor contained in each segment. In other words, the algorithm uses the feasible solution  $\chi$  to characterize the position  $x_i(t)$  in the PSO. In summary, the algorithm can be described as follows.

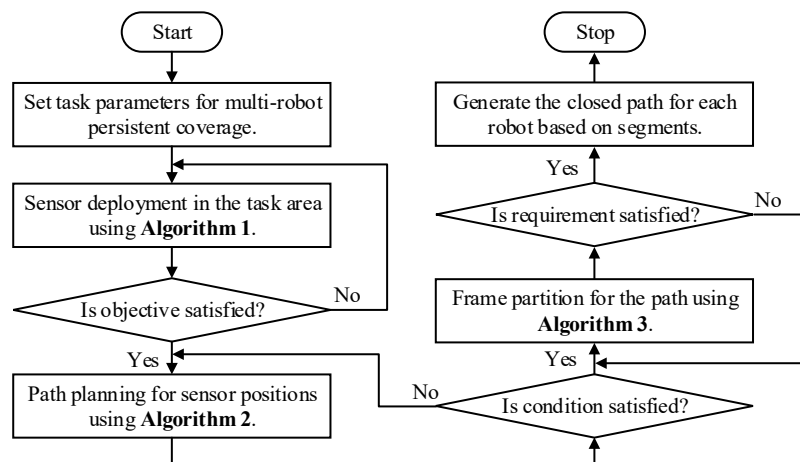


**Algorithm 3.** Pseudo-code of particle swarm optimization (PSO) for path partition.

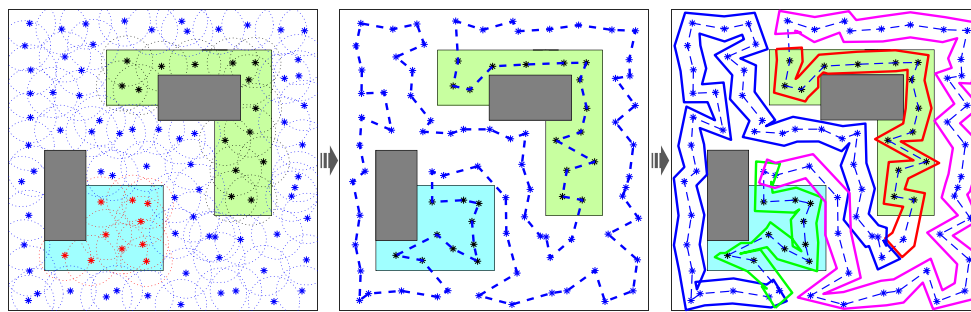
| <b>Algorithm 3:</b> Modified GA to solve TSP.                     |  |
|---|--|
| 01: // <b>Initialization:</b>                                     | 15: <b>if</b> $f_p(y_i(t)) < f_p(\hat{y}(t))$  |
| 02: Obtain the closed path from the result of <b>Algorithm 2.</b> | 16: $\hat{y}(t) = y_i(t)$  |
| 03: Set parameters of PSO.  | 17: <b>end</b>   |
| 04: Randomly initialize $K$ swarms, each with $M$ particles.      | 18: <b>end</b>   |
| 05: // <b>Main loop:</b>  | 19: Obtain the optimal solution, as well as determine the number of segments and calculate the starting and ending position of each segment. |
| 06: <b>Repeat</b> each iteration                                  | 20: Generate the closed route for each robot on the basis of portioned segment and robot's sensing range using geometry method.              |
| 07: Compute inertial weight $\omega(t)$ .                         | 21: // <b>Results:</b>   |
| 08: <b>for</b> each particle $i$                                  | 22: Get the closed route for each robot's persistent coverage.   |
| 09: The $i$ th particle is updated as (24).                       |  |
| 10: Calculate fitness value of each particle $f_p(x_i(t))$ .      |  |
| 11: <b>if</b> $f_p(x_i(t)) < f_p(y_i(t))$                         |  |
| 12: $y_i(t) = x_i(t)$   |  |
| 13: <b>end</b>  |  |
| 14: <b>end</b>  |  |

#### 4.5. The Framework of Combinatorial Method

The combinatorial method proposed in this paper, is mainly composed of the above three algorithms, namely **Algorithm 1**, **Algorithm 2**, and **Algorithm 3**. As depicted in Figure 11, the framework of combinatorial approach is proposed for multi-robot persistent coverage.

**Figure 11.** The flow chart of the combinatorial method proposed for cooperative persistent coverage.

The main principle is summarized as follows. Initially, set task parameters for persistent coverage, including coverage domains, sub-region size, obstacles' location, and so on. Then, **Algorithm 1** is used to sequentially cover the sub-regions using virtual sensors, in order of mission importance. Extract the locations of each sensor, when the entire task area is completely covered and deployment cost is minimal. Furthermore, taking the positions of the deployed sensors as the waypoints, **Algorithm 2** solves the TSP to obtain the frame of the closed path which connects all the sensors. Additionally, in consideration of coverage period constraints and the optimum quantity of robots, the frame is partitioned into several segments using **Algorithm 3**. Finally, generate the closed route for each robot on the basis of portioned segment and robot's sensing range using geometry method. Therefore, each robot can circumnavigate one closed route to cover mission domains completely and persistently. Figure 12 further illustrates the solution process of the combinatorial method for multi-robot persistent coverage.



**Figure 12.** The solution process of the combinatorial method for multi-robot persistent coverage. The cyan area is sub-region 1. The green area is sub-region 2. The white area is sub-region 3. Gray areas refer to obstacles. The asterisk denotes the position of the sensor. The blue dotted line is the frame. Solid curve represents the closed route for each robot.

## 5. Numerical Results

In view of the facts that previous works neglected regarding the coverage period constraints in different sub-regions and the number of robots used for persistent coverage task, this paper proposes the above combined method for cooperative multi-robot persistent coverage. In this section, the numerical results are presented to demonstrate the proposed approach. Simulation is implemented in Matlab. All algorithms are written by ourselves and we do not use any toolbox.

### 5.1. Multi-robot Persistent Coverage

The mission area is assumed to be a square with side length 120 m. There are two rectangular obstacles and two sub-regions with smaller coverage periods in the region of interest, as depicted in Figure 2. The coverage period constraints for each sub-region are shown in Table 1. Suppose that the velocity of each robot is a constant value of 20 m/s. Tables 2–4 list the parameters in the combinatorial scheme for cooperative persistent coverage.

**Table 1.** Coverage periods for each sub-region.

| Sub-region | Coverage Periods $T_i^c(s)$ | Sub-region | Coverage Periods $T_i^c(s)$ |
|------------|-----------------------------|------------|-----------------------------|
| 1          | 8.00                        | 3          | 40.00                       |
| 2          | 20.00                       |            |                             |

**Table 2.** Parameters of Algorithm 1 in Figure 13.

| Parameter                     | Value           | Parameter                    | Value            |
|-------------------------------|-----------------|------------------------------|------------------|
| The number of iterations      | $N_{ITE} = 150$ | Min speed of particle update | $V_{MIN} = -0.1$ |
| The number of swarms          | $N_{SWA} = 30$  | Max inertial weight          | $A_{MAX} = 0.9$  |
| Particle number of each swarm | $N_{PAR} = 50$  | Min inertial weight          | $A_{MIN} = 0.4$  |
| The number of max sensors     | $S_{MAX} = 160$ | Acceleration constant $c_1$  | $c_1 = 2$        |
| Max speed of particle update  | $V_{MAX} = 0.1$ | Acceleration constant $c_2$  | $c_2 = 2$        |

**Table 3.** Parameters of Algorithm 2 in Figure 13.

| Parameter                 | Value             | Parameter       | Value           |
|---------------------------|-------------------|-----------------|-----------------|
| The number of generations | $N_{GEN} = 10000$ | Population size | $N_{POP} = 600$ |
| The number of points      | $N_{POI} = 96$    |                 |                 |

Figure 13 illustrates the results of the path planning for multi-robot persistent coverage, in allusion to the task area situation shown in Figure 2. It can be found that the proposed method can achieve persistent coverage for the mission area using a minimum number of robots, while avoiding obstacles in complex environments. Table 5 lists the coverage periods of each partition and actual time required to circumnavigate the closed route in Figure 13.

Table 4. Parameters of Algorithm 3 in Figure 13.

| Parameter                     | Value            | Parameter                   | Value           |
|-------------------------------|------------------|-----------------------------|-----------------|
| The number of iterations      | $N_{ITE} = 50$   | Max inertial weight         | $A_{MAX} = 0.9$ |
| The number of swarms          | $N_{SWA} = 500$  | Min inertial weight         | $A_{MIN} = 0.4$ |
| Particle number of each swarm | $N_{PAR} = 1000$ | Acceleration constant $c_1$ | $c_1 = 2$       |
| Max speed of particle update  | $V_{MAX} = 0.1$  | Acceleration constant $c_2$ | $c_2 = 2$       |
| Min speed of particle update  | $V_{MIN} = -0.1$ |                             |                 |

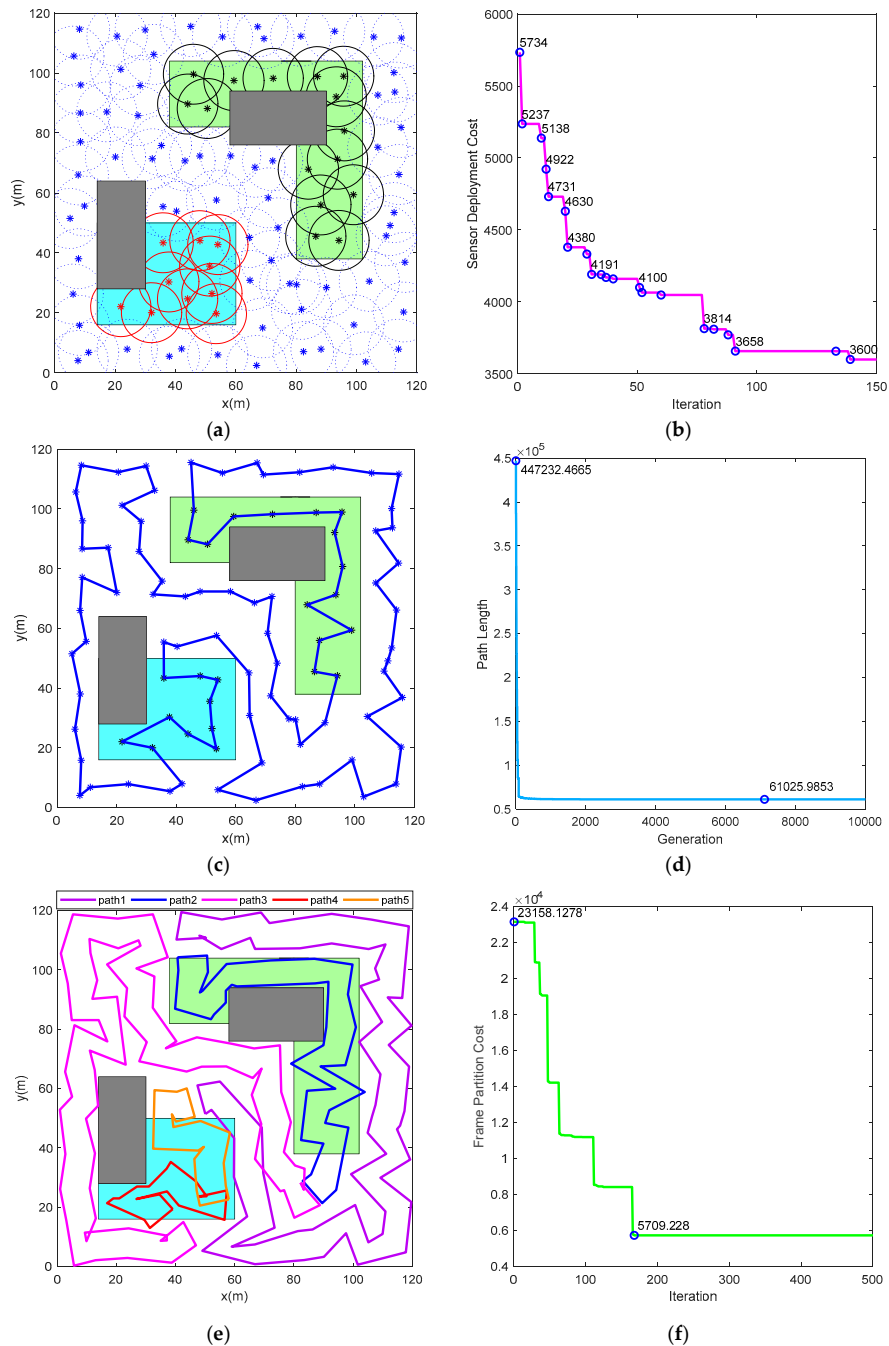


Figure 13. Simulation results under parameters in Tables 2–4. (a) Sensor deployment for complete coverage; (b) Sensor deployment cost; (c) Path planning result; (d) The changing process of path length; (e) The closed route for each robot; (f) Frame partition cost.

**Table 5.** Coverage periods and actual coverage time for each route.

| Route  | Coverage Periods $T_i^c$ (s) | Actual Coverage Time $T_i^a$ (s) |
|--------|------------------------------|----------------------------------|
| Path 1 | 40.00                        | 36.37                            |
| Path 2 | 20.00                        | 18.81                            |
| Path 3 | 40.00                        | 37.29                            |
| Path 4 | 8.00                         | 6.75                             |
| Path 5 | 8.00                         | 7.19                             |

As shown in Figure 13a, the mission area is completely covered by 96 virtual sensors, whose sensing radius is 10. Among them, there are 10 sensors in the sub-region 1, 15 sensors in the sub-region 2, and 71 sensors in the sub-region 3. Figure 13b shows the evolution of the fitness function of the sensor deployment. The optimal cost of its initial swarm is 5743. At last, the fitness value stabilized at 3600 after the 139 iterations. The path planning result for 96 waypoints is illustrated in Figure 13c. If a robot circumnavigates the closed path once, the task area can be covered completely. The history of closed path length is presented in Figure 13d. The closed path length tends to be stable after the 1722 generation, reaching the minimum of 61,025.99 by the 7113 generation. The value 61,025.99 is the minimum length calculated by (12), and the actual distance of the closed path is 1027.40. As depicted in Figure 13e, there are five closed routes generated for robots on the basis of portioned segment and robot's sensing range. Further, each closed route is assigned to one robot. It can be found that each robot can track its respective closed route to cover the task area completely and persistently. Figure 13f illustrates the changing process of cost function defined by (28). The path partition cost reaches a minimum of 5709.23 after the update of 168th iteration.

## 5.2. Simulation Results Under Different Coverage Periods

To justify the effectiveness of the proposed combinatorial scheme, this part presents some simulation results with different coverage period constraints. As depicted in Figure 14, there are six simulation results of the path planning for multi-robot persistent coverage, under different coverage period constraints. The simulation parameters used in Figure 14 are the same as in Section 5.1. Table 6 lists the coverage period  $T_i^c$  and actual time  $T_i^a$  required for each robot to circumnavigate its own closed route in allusion to different coverage period constraints. It can be found from the simulation results that the actual coverage time used of each robot satisfies the requirements of the coverage period constraints. In addition, the combined method can adaptively adjust the number of closed routes according to the coverage period constraints. Therefore, the approach can achieve optimization of the number of robots used.

There are five closed routes in Figure 14a. It is found from the Case 1 in Table 6 that each generated closed route meets the requirements of the coverage period constraints, namely  $T_i^a < T_i^c$ . As shown in Figure 13, it takes about 37.50 s for two robots to circumnavigate the closed routes in sub-region 3. Similarly, it takes 18.81 s for one robot to circumnavigate the closed routes in sub-region 2. If coverage period constraints are changed to the Case 2, the simulation result is shown in Figure 14b. In such case, the approach may generate one closed route for sub-region 2 according to the above situation. However, if this is done, the combined method will generate four closed routes for sub-region 3, because coverage period constraints of sub-region 3 in Case 2 do not satisfy the actual coverage time of sub-region 3 in Figure 13. As a result, the total routes will become six. In contrast, the technique increases the number of routes in sub-region 2 to satisfy the coverage period constraints of the task area. In this way, the total routes will become five as in Case 2. Obviously, the number of routes in Figure 14b is one less. It can be found that the combinatorial approach is able to allocate an optimal number of robots to perform persistent coverage. Figure 14c–e show the planning results for persistent coverage under corresponding coverage period constraints. It can be easily found from Figure 14e and Table 6 that the actual coverage time from the 8th route to 11th route in Case 6 is slightly higher than the coverage period constraints. The reason is that the closed routes of other regions cannot

compensate for the length of routes in sub-region 3, under the premise of satisfying their own coverage period constraints. In addition, if it is simply to meet the coverage period requirements, the cost of adding a closed route to sub-region 3 is much greater than the cost of Case 6. Therefore, the method sacrifices the coverage periods of some routes for the minimum coverage cost. In other words, the combined approach can tolerate the loss of coverage accuracy to obtain greater coverage gain.

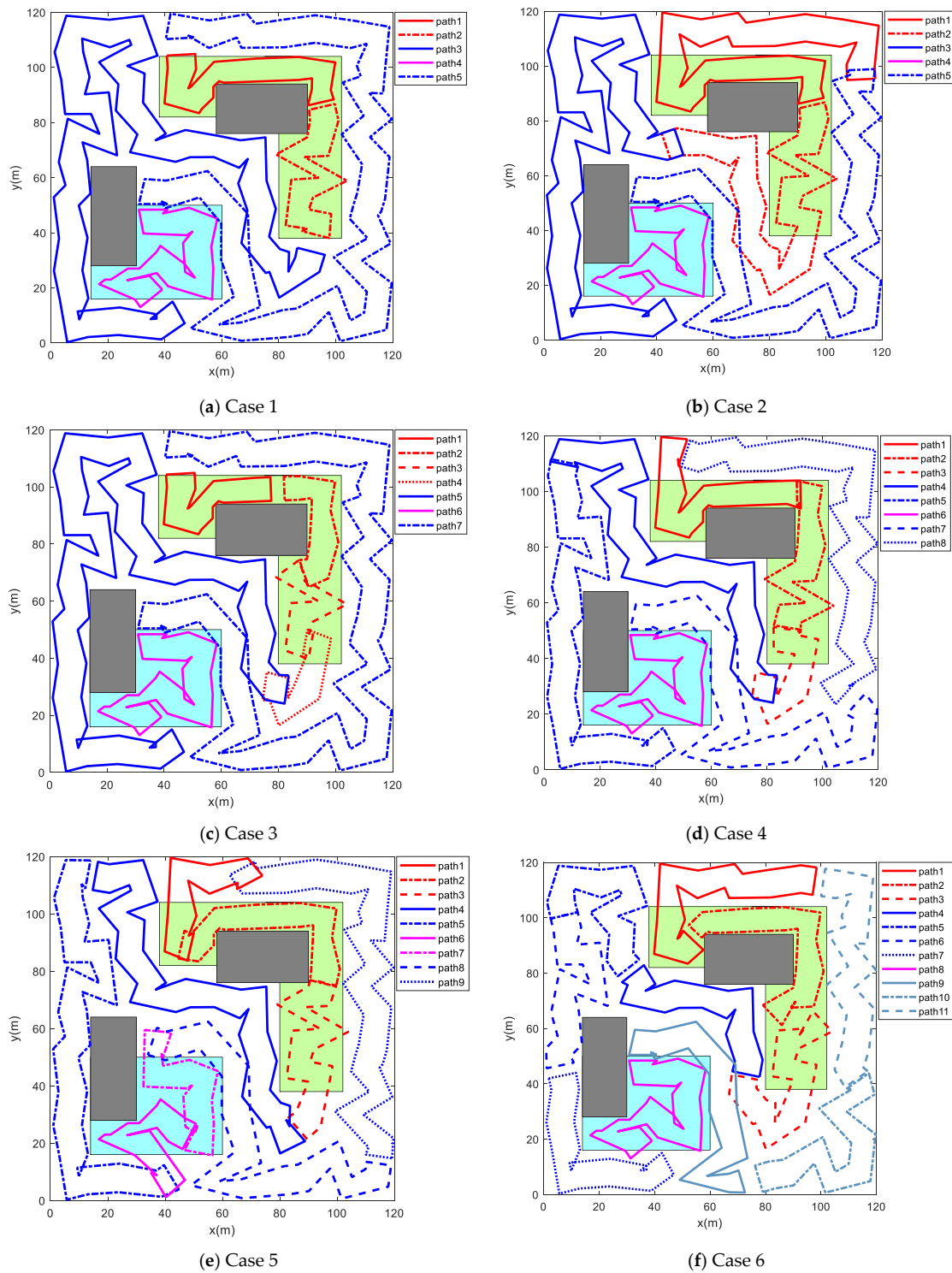


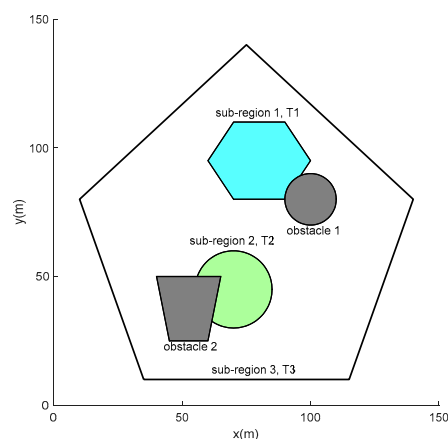
Figure 14. Simulation results under different coverage period constraints.

**Table 6.** Coverage periods and actual coverage time for each route.

| Result | Route  | Sub-region | $T_i^c(s)$ | $T_i^a(s)$ | Result | Route  | Sub-region | $T_i^c(s)$ | $T_i^a(s)$ |
|--------|--------|------------|------------|------------|--------|--------|------------|------------|------------|
| Case 1 | Path 1 | 2          | 10.00      | 9.33       | Case 2 | Path 1 | 2          | 20.00      | 19.47      |
|        | Path 2 | 2          | 10.00      | 8.68       |        | Path 2 | 2          | 20.00      | 19.41      |
|        | Path 3 | 3          | 40.00      | 38.39      |        | Path 3 | 3          | 30.00      | 28.77      |
|        | Path 4 | 1          | 12.00      | 10.95      |        | Path 4 | 1          | 12.00      | 10.95      |
|        | Path 5 | 3          | 40.00      | 38.22      |        | Path 5 | 3          | 30.00      | 29.03      |
| Case 3 | Path 1 | 2          | 7.00       | 6.30       | Case 4 | Path 1 | 2          | 10.00      | 9.33       |
|        | Path 2 | 2          | 7.00       | 5.74       |        | Path 2 | 2          | 10.00      | 8.74       |
|        | Path 3 | 2          | 7.00       | 5.91       |        | Path 3 | 2          | 10.00      | 6.35       |
|        | Path 4 | 2          | 7.00       | 5.57       |        | Path 4 | 2          | 20.00      | 18.48      |
|        | Path 5 | 3          | 40.00      | 36.22      |        | Path 5 | 3          | 20.00      | 18.83      |
|        | Path 6 | 1          | 12.00      | 10.95      |        | Path 6 | 1          | 12.00      | 10.95      |
|        | Path 7 | 3          | 40.00      | 38.22      |        | Path 7 | 3          | 20.00      | 18.92      |
|        |        |            |            | Path 8     |        | 3      | 20.00      | 18.65      |            |
| Case 5 | Path 1 | 2          | 10.00      | 6.94       | Case 6 | Path 1 | 2          | 10.00      | 9.53       |
|        | Path 2 | 2          | 10.00      | 8.82       |        | Path 2 | 2          | 10.00      | 9.75       |
|        | Path 3 | 2          | 10.00      | 9.34       |        | Path 3 | 2          | 10.00      | 8.97       |
|        | Path 4 | 3          | 20.00      | 18.28      |        | Path 4 | 3          | 10.00      | 9.05       |
|        | Path 5 | 3          | 20.00      | 19.27      |        | Path 5 | 3          | 10.00      | 9.41       |
|        | Path 6 | 1          | 8.00       | 7.24       |        | Path 6 | 3          | 10.00      | 9.05       |
|        | Path 7 | 1          | 8.00       | 7.40       |        | Path 7 | 1          | 12.00      | 10.95      |
|        | Path 8 | 3          | 20.00      | 17.20      |        | Path 8 | 3          | 10.00      | 11.64      |
|        | Path 9 | 3          | 20.00      | 18.52      |        | Path 9 | 3          | 10.00      | 10.53      |
|        |        |            |            | Path 10    |        | 3      | 10.00      | 10.90      |            |
|        |        |            |            | Path 11    |        | 3      | 10.00      | 10.41      |            |

### 5.3. Cooperative Coverage in More Complex Mission Environments

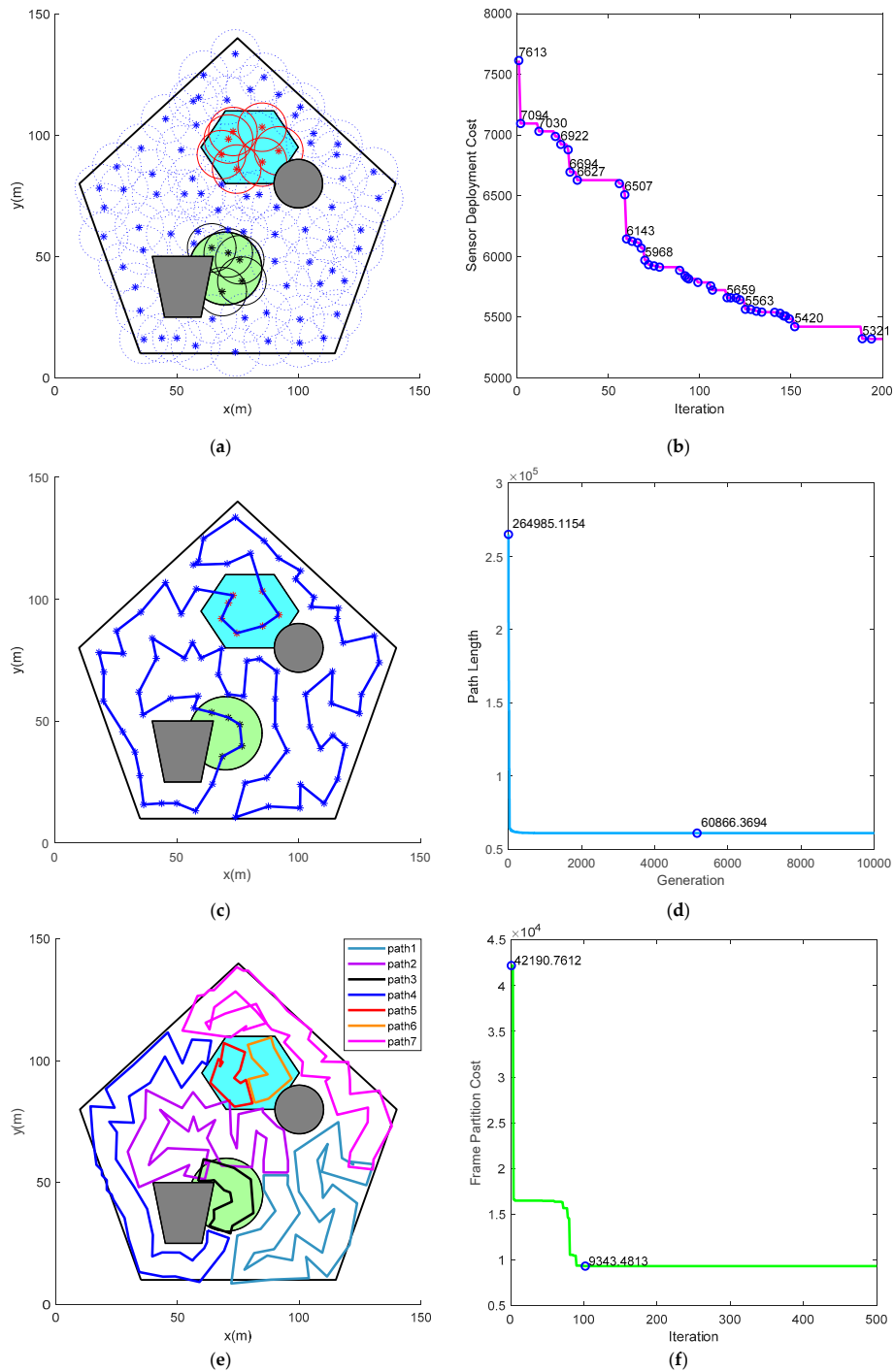
To demonstrate the versatility of the combined approach, the task area is reset using special or irregular graphics. As depicted in Figure 15, the mission area is a pentagon. There are three sub-regions and two obstacles in the task area. Sub-region 1 is a hexagonal area and sub-region 2 is a circular area. The rest of the feasible region is denoted as the sub-region 3. The coverage periods of each sub-region satisfy  $T_1 > T_2 > T_3$ .

**Figure 15.** The distribution of the task environment.

Assume that the velocity of each robot is a constant value of 10 m/s. Table 7 lists the parameters for Algorithm 1 in Figure 15. The parameters used for Algorithm 2 and 3 are the same as in Section 5.1. In allusion to the task situation shown in Figure 15, path planning results for cooperative persistent coverage are presented in Figure 16. Table 8 lists the period requirement and actual time required to circumnavigate the closed route in Figure 16.

Table 7. Parameters of Algorithm 1 in Figure 16.

| Parameter                     | Value           | Parameter                    | Value            |
|-------------------------------|-----------------|------------------------------|------------------|
| The number of iterations      | $N_{ITE} = 200$ | Min speed of particle update | $V_{MIN} = -0.1$ |
| The number of swarms          | $N_{SWA} = 40$  | Max inertial weight          | $A_{MAX} = 0.9$  |
| Particle number of each swarm | $N_{PAR} = 60$  | Min inertial weight          | $A_{MIN} = 0.4$  |
| The number of max sensors     | $S_{MAX} = 210$ | Acceleration constant $c_1$  | $c_1 = 2$        |
| Max speed of particle update  | $V_{MAX} = 0.1$ | Acceleration constant $c_2$  | $c_2 = 2$        |



**Figure 16.** Simulation results under parameters in Tables 7 and 8. (a) Sensor deployment for complete coverage; (b) Sensor deployment cost; (c) Path planning result; (d) The changing process of path length; (e) The closed route for each robot; (f) Frame partition cost.

**Table 8.** Coverage periods and actual coverage time for each route.

| Route  | $T_i^c(s)$ | $T_i^a(s)$ | Route  | $T_i^c(s)$ | $T_i^a(s)$ |
|--------|------------|------------|--------|------------|------------|
| Path 1 | 45.00      | 44.12      | Path 5 | 10.00      | 7.86       |
| Path 2 | 45.00      | 37.65      | Path 6 | 10.00      | 8.00       |
| Path 3 | 12.00      | 10.12      | Path 7 | 45.00      | 39.14      |
| Path 4 | 45.00      | 39.17      |        |            |            |

As shown in Figure 16a, the feasible region is completely covered by 83 virtual sensors. Among them, there are 7 sensors in sub-region 1, 5 sensors in sub-region 2, and 71 sensors in sub-region 3. Figure 16b describes the evolution of the fitness function of the sensor deployment. The optimal cost of its original swarm is 7613. At last, the fitness value stabilized at 5321. The path planning result for 83 waypoints is shown in Figure 16c. The history of closed path length is presented in Figure 16d. The closed path length tends to be stable after the 5156 generation, reaching the minimum of 61,025.99. As depicted in Figure 16e, there are seven closed routes generated for robots on the basis of portioned segment and robot's sensing range. Further, each closed route is allocated to one robot. Each robot can track its respective closed route to cover the task area completely and persistently. Figure 16f depicts the changing process of cost function defined by (28). The path partition cost reaches a minimum of 9343.48 after the update of the 102nd iteration. It can be found that the combinatorial method can be adapted to multi-robot persistent coverage in more complex mission environments, while guaranteeing both the obstacle avoidance and coverage period constraints.

## 6. Discussion

The main idea of the paper is to develop a new path planning strategy for multi-robot persistent coverage. In the meanwhile, the proposed scheme considers the coverage period constraints of different sub-regions in the mission domains, and can achieve the collaborative persistent coverage in more complex mission environments using an optimal number of the robots, while guaranteeing both obstacle avoidance and coverage period constraints.

On the basis of the theoretical analysis and numerical simulation, the solution strategy of proposed method can be divided into three steps. The first step is to cover the task area completely using the virtual sensors with minimum cost. According to the coverage periods of sub-regions from small to large, sensor deployment in the feasible region can be performed in several stages. The sub-region with the smallest coverage period is primarily deployed, and then the sub-region with the larger period is deployed until all the sub-regions are completely covered. After determining the coverage order, the sub-regions are sequentially covered by using **Algorithm 1**. The sensor deployment of each sub-region is optimized independently to reduce the complexity of the algorithm and speed up the convergence process. When the feasible area is completely covered, the second step is to obtain the frame by using **Algorithm 2**. Taking the position of the deployed sensor as the waypoint, a closed path connecting all waypoints can be acquired by solving the TSP. Moreover, the shortest path that connects any two waypoints is obtained based on the A-star algorithm. The proposed approach introduces the idea of sensor deployment to divides path planning issues into SDP and TSP in order to effectively cope with the planning puzzle caused by environmental obstacles. Simulation results show that the method can effectively improve the adaptability and robustness of the algorithm to the more complex environment. Given the closed frame, the third step is to generate the closed route for each robot by using **Algorithm 3**. Before generating the closed routes, it is first necessary to divide the path frame into several segments on the basis of the coverage period constraints. Actually, the purpose of the frame partition is to divide the frame into the least number of segments under the premise of satisfying the coverage period constraints. Afterwards, the computational geometry technique is utilized to generate the closed route for each robot on the basis of each partitioned segment. Finally, each closed route is allocated to one robot to circumnavigate periodically in order to achieve persistent coverage of the feasible region. Numerical results indicate that the combinatorial approach is able to



allocate an optimal number of robots to perform persistent coverage and can adaptively adjust the number of robots used according to the coverage period constraints. In conclusion, the persistent coverage achieved in this paper has the following characteristics: (1) cooperative persistent coverage for multiple robots is achieved; (2) the planning puzzle caused by environmental obstacles is solved; (3) the coverage period constraints of different sub-regions are considered; (4) the number of robots used can be adaptively adjust according to the coverage period constraints; (5) the optimal number of robots is utilized to perform the persistent coverage task.

Similar works for multi-robot cooperative coverage can be found in [13,46]. Karapetyan et al. [13] present two approximation heuristics based on boustrophedon cellular decomposition for solving the multi-robot complete coverage. The first algorithm is a direct extension of the work of Xu et al. [10] for multi-robot systems. The solution process can be divided into three steps. The approach first partitions the task area into nonoverlapping cells. Then it solves the Chinese postman problem to find a single optimal route that covers these cells. Finally, the algorithm splits the resulting route between multiple robots using the k-postman approximation algorithm proposed by [47]. Different from the first method, the second scheme first divides the task area into approximately equal partitions between robots by using greedy approach, then it utilizes the coverage algorithm proposed by [10] to plan the coverage route for each sub-region. The algorithms proposed by Karapetyan et al. can commendably solve the problem of area decomposition and route planning for complex environments with obstacles, and provide a new solution for collaborative coverage problems in complex task areas. Compared with the combinatorial approach proposed in this paper, the works of Karapetyan et al. mainly have the following differences: (1) the methods proposed by [13] achieve the complete coverage of the given area, not a periodic persistent coverage; (2) the number of robots performing cooperative coverage is artificially determined and not adaptively adjusted according to task requirements; (3) the sub-regions with different task importance in the mission area are not considered, similarly the coverage period constraints of different sub-regions are neglected. Palacios-Gasós et al. [46] develop an online algorithm for multi-robot persistent coverage in which each robot locally finds the optima paths and coverage actions to maintain the desired coverage level over the whole area. Firstly, the method divides the task area into several particular regions by using Voronoi diagrams, one for each robot, to avoid long shifts and conflicts with the other robots. Then, each robot creates a list of potential goals that includes the points of its region in which the coverage level can be improved the most. Next, the algorithm calculates the candidate paths to all potential goals from the list using the fast-marching method. Finally, the optimal path is selected to calculate the optimal coverage action and control the movement of the robot. The distributed algorithm proposed by [46] can actively select the coverage goals in a continuous environment and plan the optimal paths to such goals. Furthermore, the dynamic window approach for navigation is introduced to efficiently improve the algorithm competitive in terms of flexibility and robustness in changing environments. The work of Palacios-Gasós et al. can implement multi-robot cooperative persistent coverage effectively and reach the requirement of the desired coverage level quickly. Similarly, the method proposed by [46] neglects the coverage period constraints of the sub-regions with different task importance in the mission area. Moreover, the number of robots used is also preset and not adaptively adjusted according to task requirements. However, Palacios-Gasós et al. present a novel path planning for solving multi-robot persistent coverage in complex task environments.

The proposed method in this paper is an offline method for multi-robot persistent coverage and lacks experiment results. Palacios-Gasós et al. present a new online algorithm for solving multi-robot persistent coverage in complex task areas. The future work will focus on the online path planning for cooperative persistent coverage with reference to the work of Palacios-Gasós et al. and give the actual experiments for proving the effectiveness of the proposed approach. In addition, future research will be also centered on the planning of dynamic sub-regions, namely, the sub-region and coverage period are not fixed by the prior information. Furthermore, we will promote the proposed algorithm by using the more advanced machine learning algorithms [48,49] for multi-robot persistent coverage.

## 7. Conclusions

This paper presents a new combinatorial method for multi-robot persistent coverage in complex mission environments using an optimal number of robots.

(1) The proposed method achieves path planning for cooperative persistent coverage, in complex task areas. The path planning problem is decomposed into the sensor deployment problem and the travelling salesman problem. The planning technique, based on sensor deployment, effectively solves the obstacle avoidance in a complex environment.

(2) Sub-regions with different task importance in the mission area are considered in this paper. Moreover, the combined method can adaptively adjust the number of closed routes according to the coverage period constraints of different sub-regions, while also optimizing the number of robots performing the task.

According to the aforementioned description of the combinatorial approach, the planning results of the proposed method are highly effective for solving the cooperative persistent coverage. From the global perspective, the design approach takes the coverage period as the main basis and generates the closed routes for each robot in terms of the mission environment. Furthermore, each robot can circumnavigate one closed route to cover task domains completely and persistently.

**Author Contributions:** Conceptualization, R.Z. and G.S.; Methodology, G.S. and B.D.; Software, G.S. and B.D.; Validation, R.Z., Z.D. and Y.W.; Formal Analysis, G.S. and Z.D.; Investigation, G.S. and Z.D.; Resources, R.Z. and Y.W.; Data Curation, Z.D.; Writing—Original Draft Preparation, R.Z. and G.S.; Writing—Review and Editing, G.S. and B.D.; Visualization, G.S. and B.D.; Supervision, R.Z. and Z.D.; Project Administration, R.Z. and Z.D.; Funding Acquisition, R.Z. and Z.D.

**Funding:** This research was funded by the National Natural Science Foundation of China under Grant 61773031 and Grant 61573042.

**Conflicts of Interest:** The authors declare no conflict of interest.

## References

1. Chen, S.; Wu, F.; Shen, L.; Chen, J.; Ramchurn, S.D. Decentralized patrolling under constraints in dynamic environments. *IEEE Trans. Cybern.* **2016**, *46*, 3364–3376. [[CrossRef](#)]
2. Li, P.; Duan, H. A potential game approach to multiple UAV cooperative search and surveillance. *Aerosp. Sci. Technol.* **2017**, *68*, 403–415. [[CrossRef](#)]
3. Masehian, E.; Jannati, M.; Hekmatfar, T. Cooperative mapping of unknown environments by multiple heterogeneous mobile robots with limited sensing. *Robot. Autom. Syst.* **2017**, *87*, 188–218. [[CrossRef](#)]
4. Paull, L.; Seto, M.; Leonard, J.J.; Li, H. Probabilistic cooperative mobile robot area coverage and its application to autonomous seabed mapping. *Int. J. Robot. Res.* **2018**, *37*, 21–25. [[CrossRef](#)]
5. Razi, P.; Sumantyo, J.T.S.; Perissin, D.; Kuze, H.; Chua, M.Y.; Panggabean, G.F. 3D land mapping and land deformation monitoring using persistent scatterer interferometry (PSI) ALOS PALSAR: validated by geodetic GPS and UAV. *IEEE Access* **2018**, *6*, 12395–12404. [[CrossRef](#)]
6. Huang, L.; Qu, H.; Zou, L. Multi-type UAVs cooperative task allocation under resource constraints. *IEEE Access* **2018**, *6*, 17841–17850. [[CrossRef](#)]
7. Portugal, D.; Rocha, R.H. Distributed multi-robot patrol: A scalable and fault-tolerant framework. *Robot. Autom. Syst.* **2013**, *61*, 1572–1587. [[CrossRef](#)]
8. Nigam, N.; Bieniawski, S.; Kroo, I.; Vian, J. Control of multiple UAVs for persistent surveillance: algorithm and flight test results. *IEEE Trans. Control Syst. Technol.* **2012**, *20*, 1236–1251. [[CrossRef](#)]
9. Peters, J.R.; Wang, S.J.; Surana, A.; Bullo, F. Cloud-supported coverage control for persistent surveillance missions. *J. Dyn. Syst. Meas. Contr.* **2017**, *139*, 1–12. [[CrossRef](#)]
10. Xu, A.; Viriyasuthee, C.; Rekleitis, I. Efficient complete coverage of a known arbitrary environment with applications to aerial operations. *Auton. Robots* **2014**, *36*, 365–381. [[CrossRef](#)]
11. Mansouri, S.S.; Kanellakis, C.; Georgoulas, G.; Kominiak, D.; Gustafsson, T.; Nikolakopoulos, G. 2D visual area coverage and path planning coupled with camera footprints. *Control Eng. Pract.* **2018**, *75*, 1–16. [[CrossRef](#)]

12. Galceran, E.; Carreras, M. A survey on coverage path planning for robotics. *Robot. Autom. Syst.* **2013**, *61*, 1258–1276. [[CrossRef](#)]
13. Karapetyan, N.; Benson, K.; McKinney, C.; Taslakian, P.; Rekleitis, I. Efficient multi-robot Coverage of a Known Environment. In Proceedings of the 2017 IEEE International Conference on Intelligent Robots and Systems, Vancouver, BC, Canada, 24–28 September 2017.
14. Koenig, S.; Szymanski, B.; Liu, Y. Efficient and inefficient ant coverage methods. *Ann. Math. Artif. Intell.* **2001**, *31*, 41–77. [[CrossRef](#)]
15. Votion, J.; Cao, Y. Diversity-based cooperative multivehicle path planning for risk management in costmap environments. *IEEE Trans. Ind. Electron.* **2019**, *66*, 6117–6127. [[CrossRef](#)]
16. Ranjitha, T.D.; Guruprasad, K.R. Pseudo spanning tree-based complete and competitive robot coverage using virtual nodes. *IFAC-Pap. Online* **2016**, *491*, 185–200. [[CrossRef](#)]
17. Elmaliach, Y.; Agmon, N.; Kaminka, G.A. Multi-robot area patrol under frequency constraints. *Ann. Math. Artif. Intell.* **2009**, *57*, 293–320. [[CrossRef](#)]
18. Franco, C.; Vachtsevanos, G.; Tang, L. Multi-unmanned aerial vehicle coverage planner for area surveillance missions. In Proceedings of the AIAA Guidance, Navigation and Control Conference and Exhibit, Hilton Head, SC, USA, 20–23 August 2007.
19. Franco, C.; Stipanović, D.M.; López-Nicolás, G.; Sagüés, C.; Llorente, S. Persistent coverage control for a team of agents with collision avoidance. *Eur. J. Control* **2015**, *22*, 30–45. [[CrossRef](#)]
20. Erignac, C.A. An exhaustive swarming search strategy based on distributed pheromone maps. In Proceedings of the AIAA Infotech@Aerospace 2007 Conference and Exhibit, Rohnert Park, CA, USA, 7–10 May 2007.
21. Lima, D.A.; Oliveira, M.B. A cellular automata ant memory model of foraging in a swarm of robots. *Appl. Math. Model.* **2017**, *47*, 551–572. [[CrossRef](#)]
22. Liu, R.; Li, J.; Mu, C.; Jiao, L. A coevolutionary technique based on multi-swarm particle swarm optimization for dynamic multi-objective optimization. *Eur. J. Oper. Res.* **2017**, *261*, 1028–1051. [[CrossRef](#)]
23. Tumer, T.; Agogino, A. Coordinating multi-rover systems: Evaluation functions for dynamic and noisy environments. In Proceedings of the 7th annual conference on Genetic and evolutionary computation, Washington, DC, USA, 25–29 June 2005.
24. Hokayem, P.F.; Stipanović, D.; Spong, M.W. On persistent coverage control. In Proceedings of the 46th IEEE Conference on Decision and Control, New Orleans, LA, USA, 12–14 December 2007.
25. Cui, R.; Li, Y.; Yan, W. Mutual information-based multi-AUV path planning for scalar field sampling using multidimensional RRT\*. *IEEE Trans. Syst. Man Cybern.* **2016**, *46*, 993–1003. [[CrossRef](#)]
26. Atınç, G.M.; Stipanović, D.M.; Voulgaris, P.G. Supervised coverage control of multi-agent systems. *Automatica* **2014**, *50*, 2936–2942. [[CrossRef](#)]
27. Song, C.; Liu, L.; Feng, G.; Wang, Y.; Gao, Q. Persistent awareness coverage control for mobile sensor networks. *Automatica* **2013**, *49*, 1867–1873. [[CrossRef](#)]
28. Smith, S.L.; Rus, D. Multi-robot monitoring in dynamic environments with guaranteed currency of observations. In Proceedings of the 49th IEEE Conference on Decision and Control, Atlanta, GA, USA, 15–17 December 2010.
29. Stump, E.; Michael, N. Multi-robot persistent surveillance planning as a vehicle routing problem. In Proceedings of the 2011 IEEE International Conference on Automation Science and Engineering, Trieste, Italy, 24–27 August 2011.
30. Mansouri, S.S.; Kanellakis, C.; Fresk, E.; Kominiak, D.; Nikolakopoulos, G. Cooperative coverage path planning for visual inspection. *Control Eng. Pract.* **2017**, *74*, 118–131. [[CrossRef](#)]
31. Volos, C.K.; Kyprianidis, I.M.; Stouboulos, I.N. Experimental investigation on coverage performance of a chaotic autonomous mobile robot. *Robot. Autom. Syst.* **2013**, *61*, 1314–1322. [[CrossRef](#)]
32. Karatas, M. A multi-objective facility location problem in the presence of variable gradual coverage performance and cooperative cover. *Eur. J. Oper. Res.* **2017**, *262*, 1040–1051. [[CrossRef](#)]
33. Ellefsen, K.O.; Lepikson, H.A.; Albiez, J.C. Multi-objective coverage path planning: Enabling automated inspection of complex, real-world structures. *Appl. Soft Comput.* **2017**, *61*, 264–282. [[CrossRef](#)]
34. Wang, X.; Wang, S.; Ma, J.J. An improved co-evolutionary particle swarm optimization for wireless sensor networks with dynamic deployment. *Sensors* **2007**, *7*, 354–370. [[CrossRef](#)]
35. Li, X.D.; Yao, X. Cooperatively coevolving particle swarms for large scale optimization. *IEEE Trans. Evol. Comput.* **2012**, *16*, 210–224.

36. Wang, Y.C.; Tseng, Y.C. Distributed deployment schemes for mobile wireless sensor networks to ensure multilevel coverage. *IEEE Trans. Parallel Distrib. Syst.* **2008**, *19*, 1280–1294. [[CrossRef](#)]
37. Chang, C.Y.; Chang, C.T.; Chen, Y.C.; Chang, H.R. Obstacle-resistant deployment algorithms for wireless sensor networks. *IEEE Trans. Veh. Technol.* **2009**, *58*, 2925–2941. [[CrossRef](#)]
38. Seok, J.H.; Lee, J.Y.; Kim, W.; Lee, J.J. A bipopulation-based evolutionary algorithm for solving full area coverage problems. *IEEE Sens. J.* **2013**, *13*, 4796–4807. [[CrossRef](#)]
39. Di, B.; Zhou, R.; Zhang, Y.; Che, J. Path planning for unmanned vehicle searching based on sensor deployment and travelling salesman problem. In Proceedings of the 2014 IEEE Chinese Guidance, Navigation and Control Conference, Yantai, China, 8–10 August 2014.
40. Li, F.; Luo, J.; Xin, S.; He, Y. Autonomous deployment of wireless sensor networks for optimal coverage with directional sensing model. *Comput. Netw.* **2016**, *108*, 120–132. [[CrossRef](#)]
41. AlKaraki, J.N.; Gawamneh, A. The optimal deployment, coverage, and connectivity problems in wireless sensor networks: revisited. *IEEE Access* **2017**, *5*, 18051–18065. [[CrossRef](#)]
42. Fang, W. Novel efficient deployment schemes for sensor coverage in mobile wireless sensor networks. *Inf. Fusion* **2018**, *41*, 25–38. [[CrossRef](#)]
43. Maza, I.; Ollero, A. Multiple UAV cooperative searching operation using polygon area decomposition and efficient coverage algorithms. In Proceedings of the 7 th International Symposium on Distributed Autonomous Robotic Systems, Toulouse, France, 23–25 June 2007.
44. Ghosh, A.; Das, S.K.; Lee, J.J. Coverage and connectivity issues in wireless sensor networks: A survey. *Pervasive Mob. Comput.* **2008**, *4*, 303–334. [[CrossRef](#)]
45. Zhan, A.H.; Zhang, J.; Li, Y.; Shi, Y.H. Orthogonal learning particle swarm optimization. *IEEE Trans. Evol. Comput.* **2011**, *15*, 832–847. [[CrossRef](#)]
46. Palacios-Gasós, J.M.; Talebpour, Z.; Montijano, E.; Sagüés, C.; Martinoli, A. Optimal path planning and coverage control for multi-robot persistent coverage in environments with obstacles. In Proceedings of the 2017 International Conference on Robotics and Automation, Singapore, 29 May–3 June 2017.
47. Frederickson, G.N.; Hecht, M.S.; Kim, C.E. Approximation algorithms for some routing problems. In Proceedings of the 17th Annual Symposium on Foundations of Computer Science, Houston, TX, USA, 25–27 October 1976.
48. Castaño, F.; Beruvides, C.; Haber, R.E.; Artuñedo, A. Obstacle recognition based on machine learning for on-chip LiDAR sensors in a cyber-physical system. *Sensors* **2017**, *17*, 2109. [[CrossRef](#)] [[PubMed](#)]
49. Castaño, F.; Beruvides, C.; Villalonga, A.; Haber, R.E. Self-tuning method for increased obstacle detection reliability based on internet of things LiDAR sensor models. *Sensors* **2018**, *18*, 1508. [[CrossRef](#)] [[PubMed](#)]



© 2019 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<http://creativecommons.org/licenses/by/4.0/>).